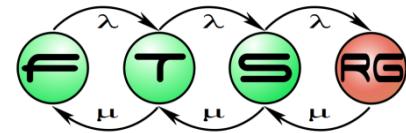


# Doman-specific modeling based on the Eclipse Modeling Framework



# Motivation

- Modern software/system development
  - High complexity
    - Increasing development time
    - Increasing cost
  - Diversity
    - Domain
    - Requirements
    - Implementation tools and techniques
  - Changes
    - Environment
    - Requirements
    - Bug fixes

# General Purpose Languages

- Multiple languages
  - C, Java, Javascript, Go, Ruby, Python, ...
- Capable of solving all problems (Turing-complete)
- Domain expert cannot efficiently use it
  - Programming knowledge needed!

# Domain-specific language

A DSL is a **focused, processable** language for describing a specific concern when building a system in a specific domain. The **abstractions** and **notations** used are natural/suitable for the **stakeholders** who specify that particular concern.

Markus Voelter: DSL Design

|                                          | GPL                             | DSL                           |
|------------------------------------------|---------------------------------|-------------------------------|
| <b>Domain</b>                            | Large and complex               | Smaller and well-defined      |
| <b>Language Size</b>                     | Large                           | Small                         |
| <b>Turing completeness</b>               | Always                          | Often not                     |
| <b>User-defined abstractions</b>         | Sophisticated                   | Limited                       |
| <b>Execution</b>                         | Via intermediate GPL            | Native                        |
| <b>Lifespan</b>                          | Years to decades                | Months to years               |
| <b>Designed by</b>                       | Guru or committee               | Few engineers, domain experts |
| <b>User community</b>                    | Large, anonymous and widespread | Small, accessible and local   |
| <b>Evolution</b>                         | Slow, often standardized        | Fast-paced                    |
| <b>Deprecation, incompatible changes</b> | Almost impossible               | Feasible                      |

# Evaluation of DSLs

## ■ Advantages

- Quick solutions to specific problems
- Requirements are expressed on the problem domain
- Automatic (requirement) validation possible

## ■ Disadvantages

- Domain knowledge required (can be expensive)
- Language design can be subjective and/or complex
- Tooling cost not negligible (wrt limited focus)
- Incompatible DSL's for the same domain
  - Standardization may be required

# Language-focused development

- Language
  - Describes a model
  - For **humans**, not compiler
  - Gives domain-level overview
- Tooling required for
  - Execution
  - Analysis
  - Testing
  - Editing
  - ...

# Multi-language aspects

- Multi-language development
  - Transformation
  - Aspect integration
  - Synchronization
  - Refinement or equivalence checking
  - Evolution
- Similar to aspect-oriented programming

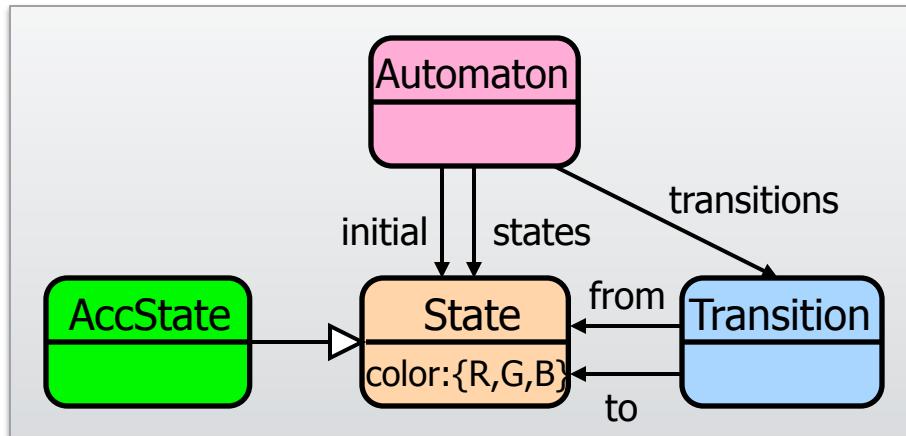
# Metamodeling

Design paradigm for modeling languages

# Designing modeling languages

- Metamodel: a model of models
  - Abstract syntax
  - Concrete syntax
  - Well-formedness rules
  - Behavioral (dynamic) semantics
  - Translation to other languages

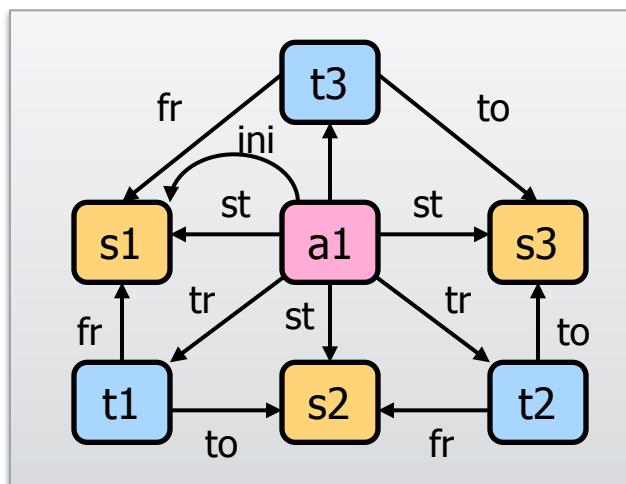
# Meta- and instance models



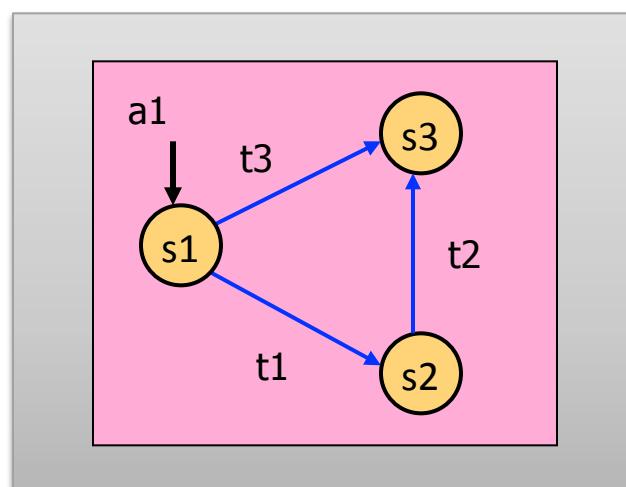
Metamodel level

Metamodel

Model level

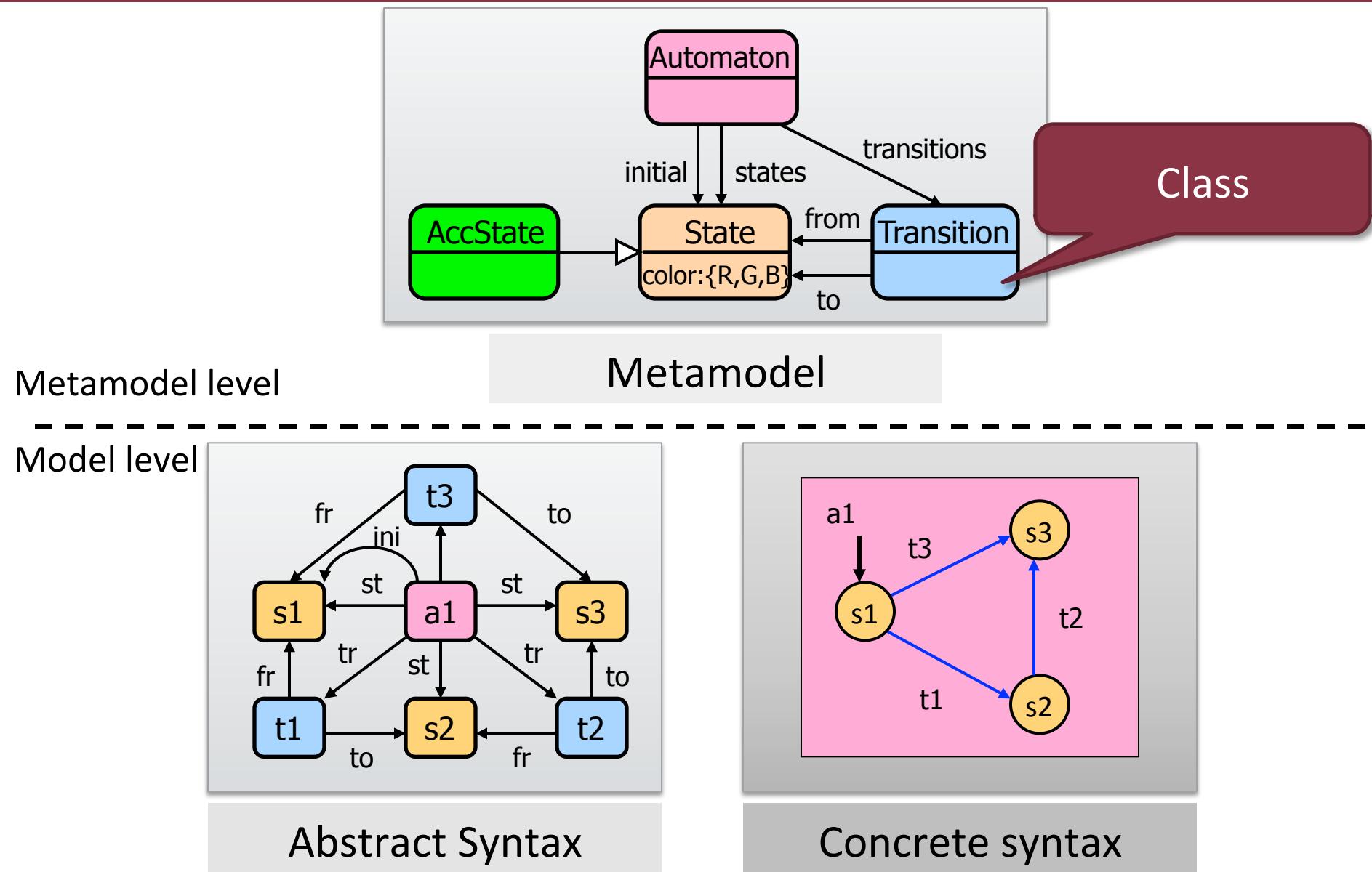


Abstract Syntax

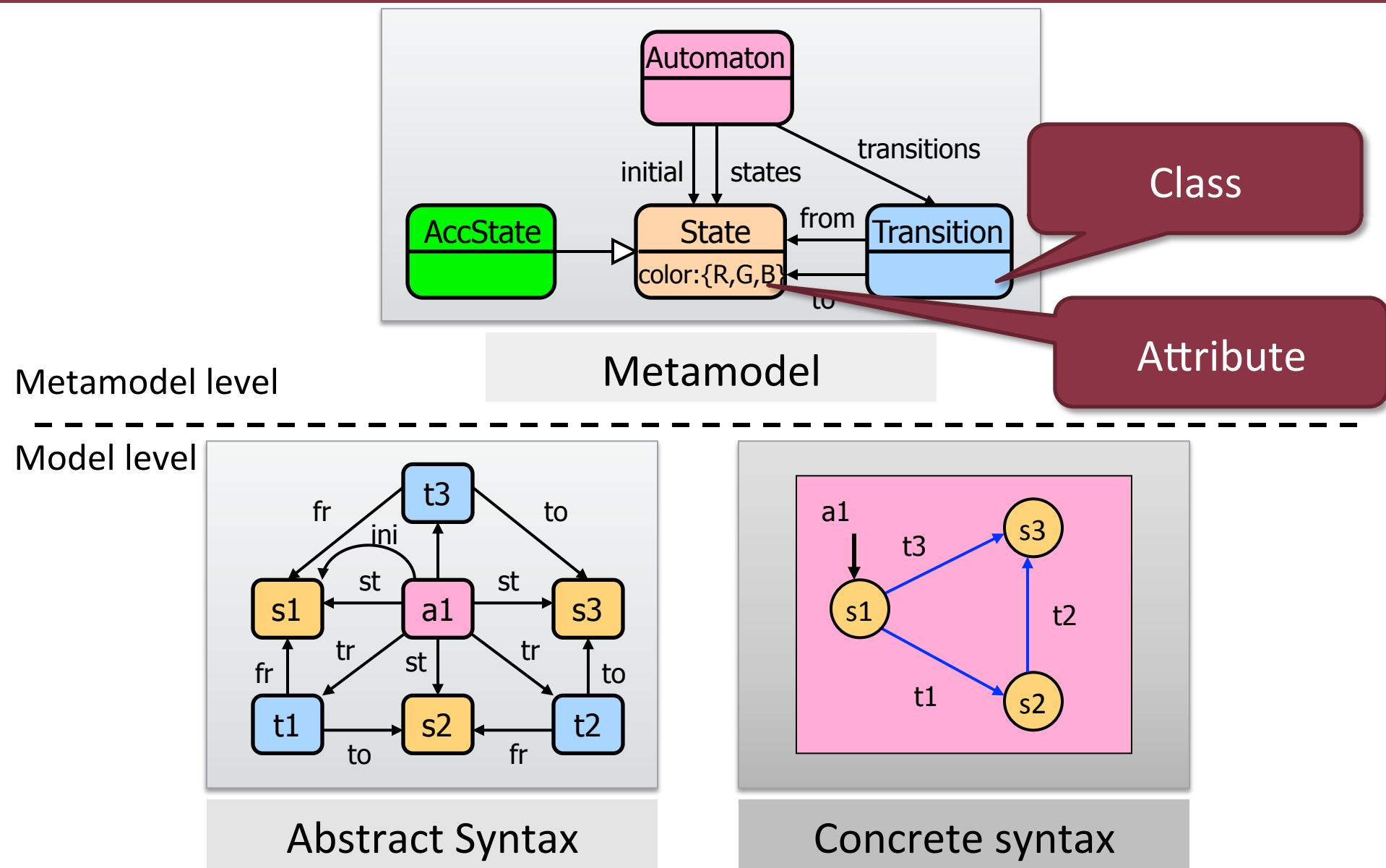


Concrete syntax

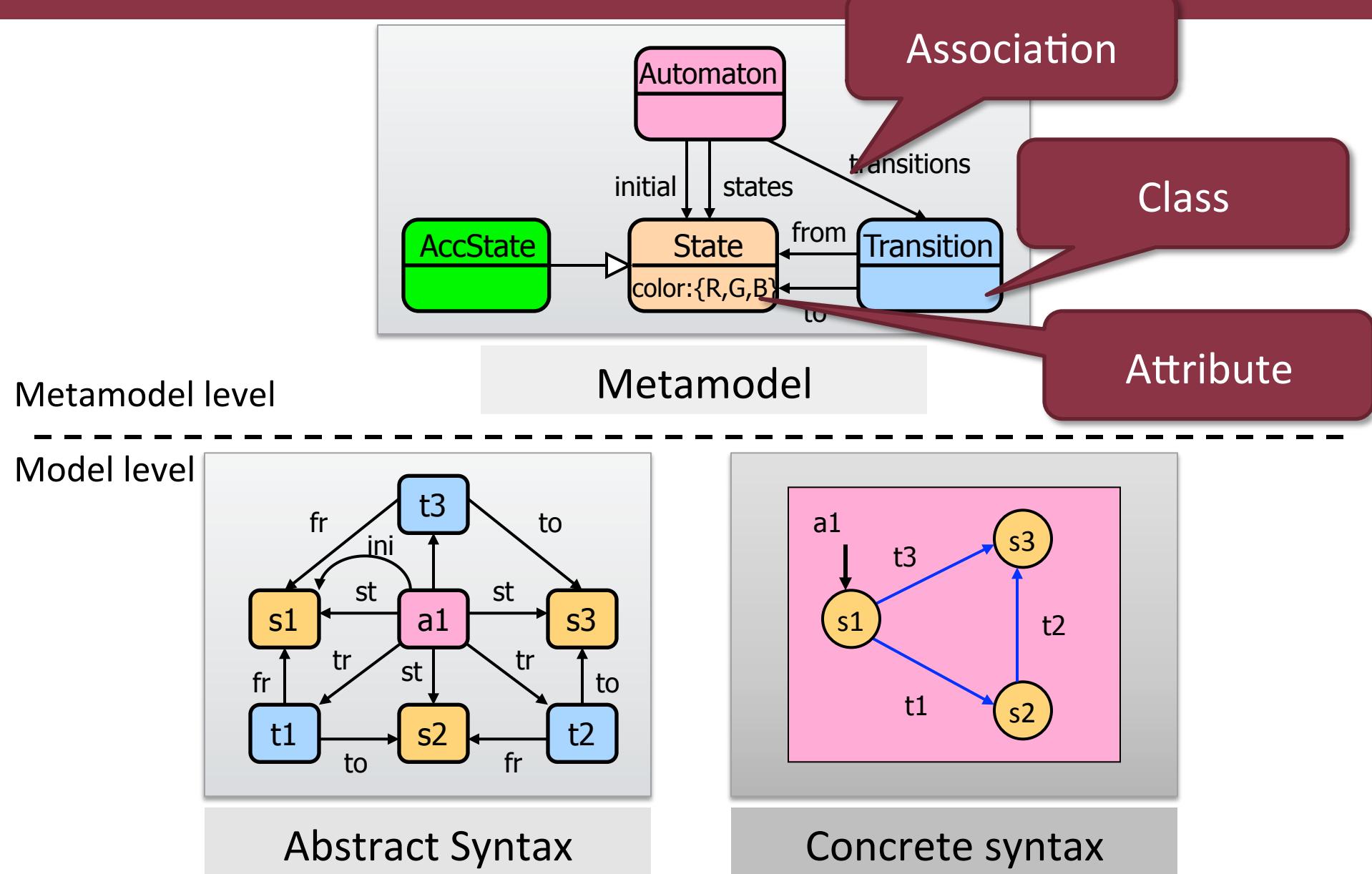
# Meta- and instance models



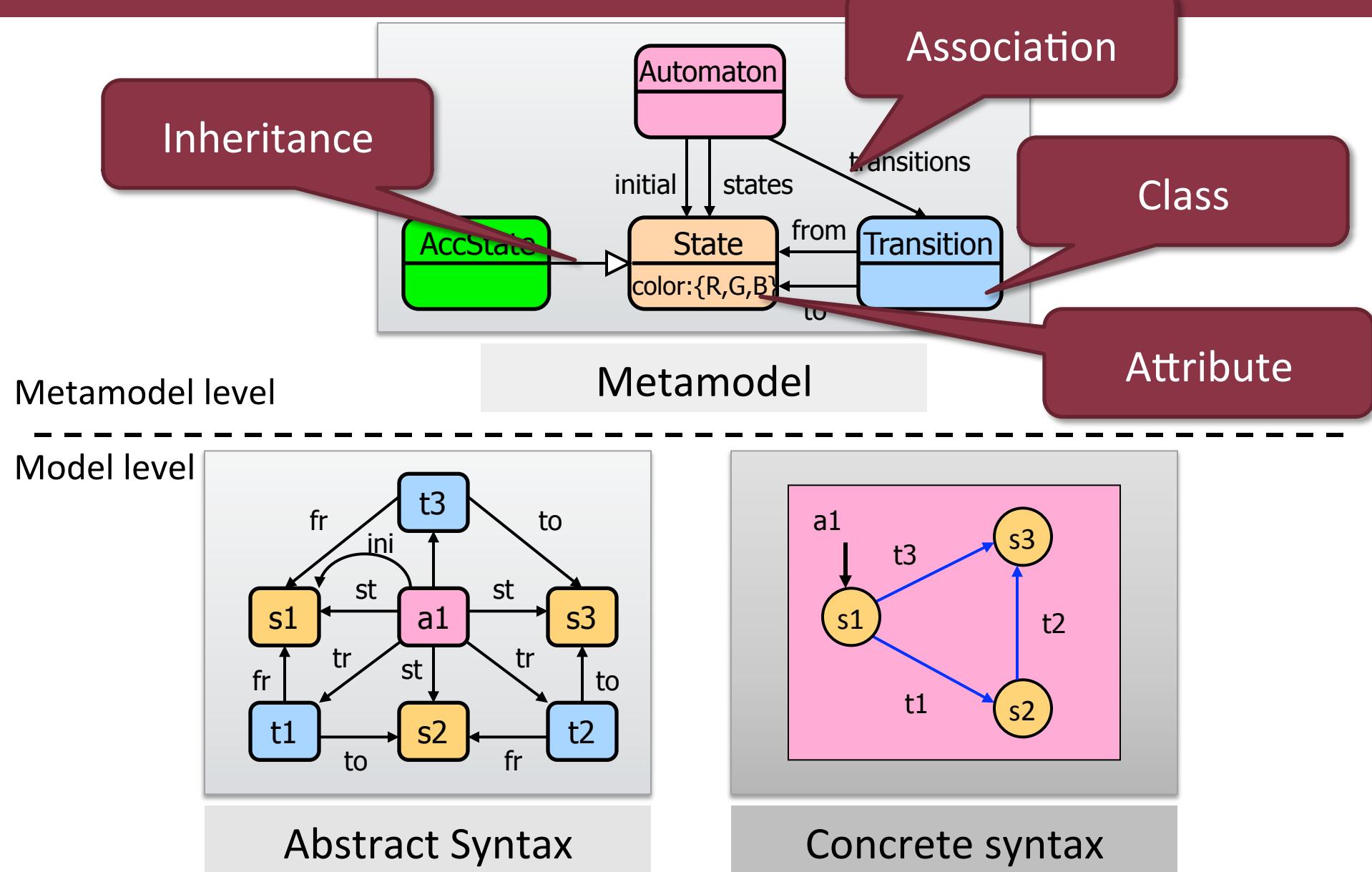
# Meta- and instance models



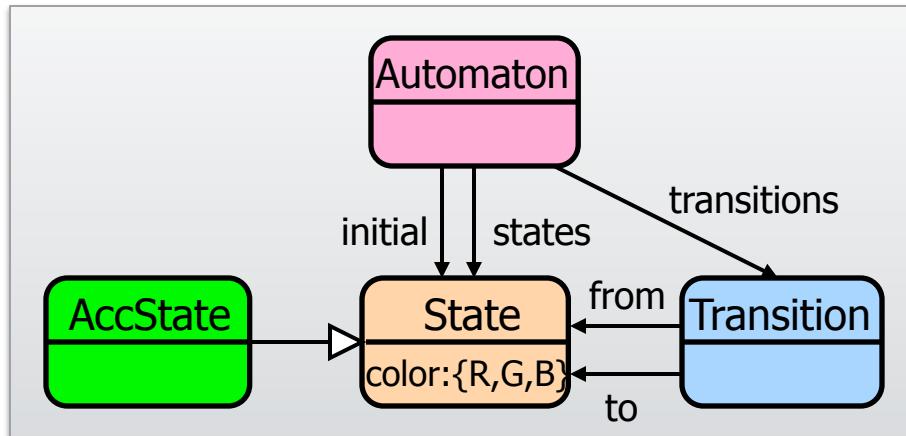
# Meta- and instance models



# Meta- and instance models



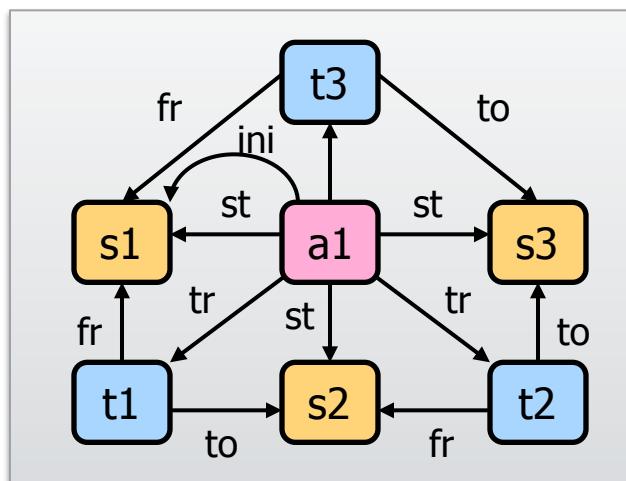
# Meta- and instance models



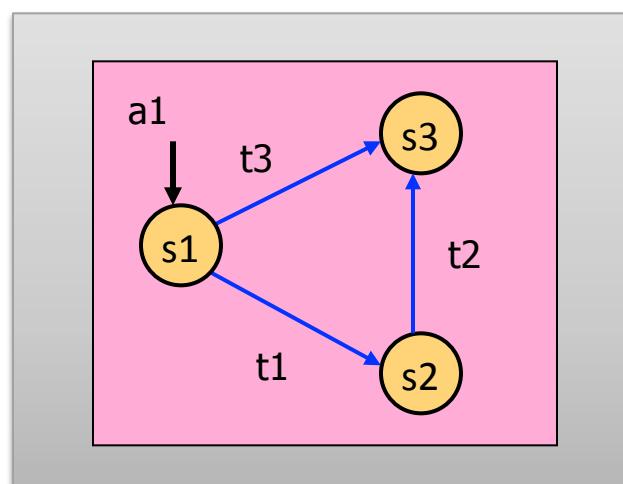
Metamodel level

Metamodel

Model level

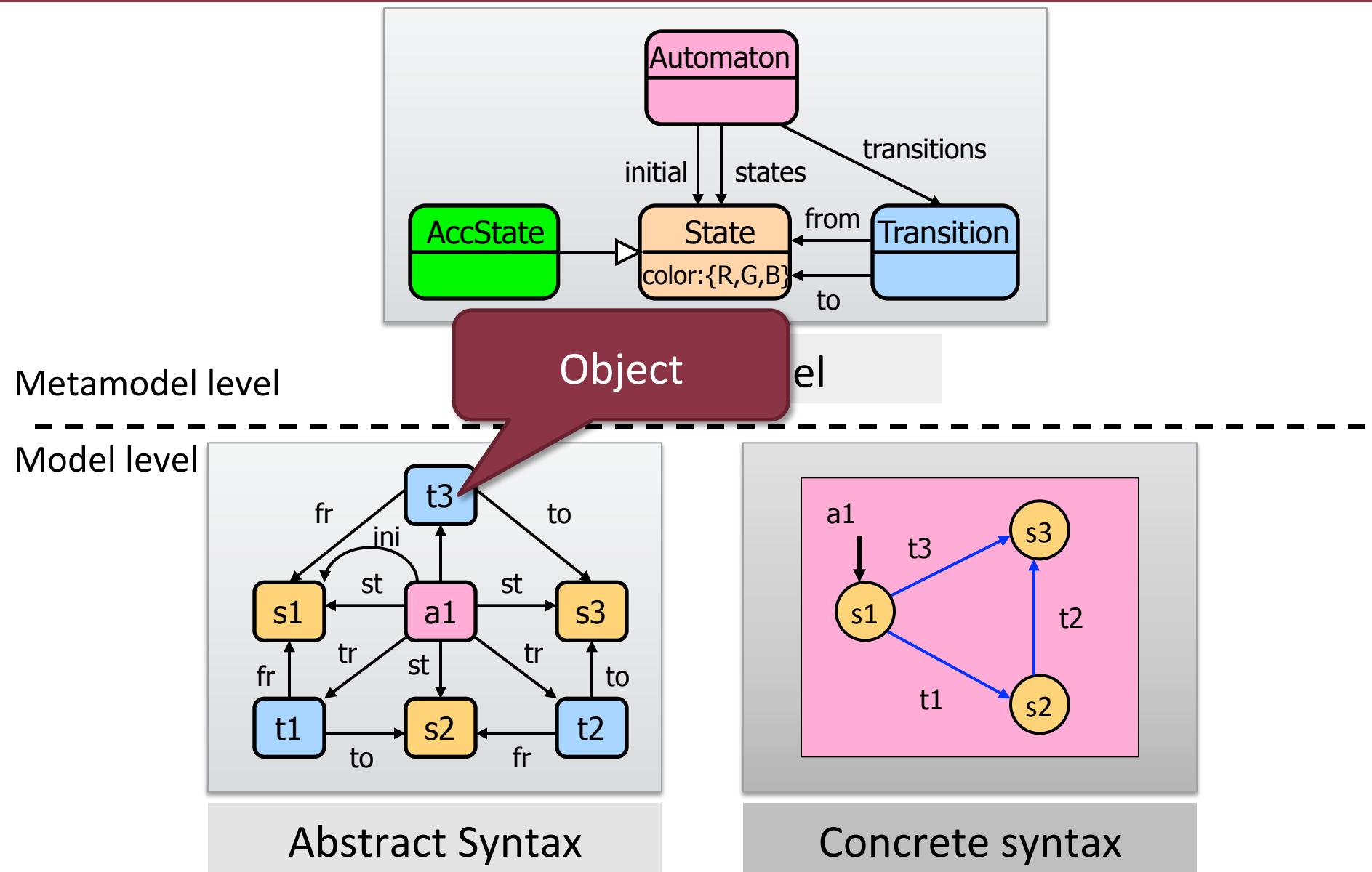


Abstract Syntax

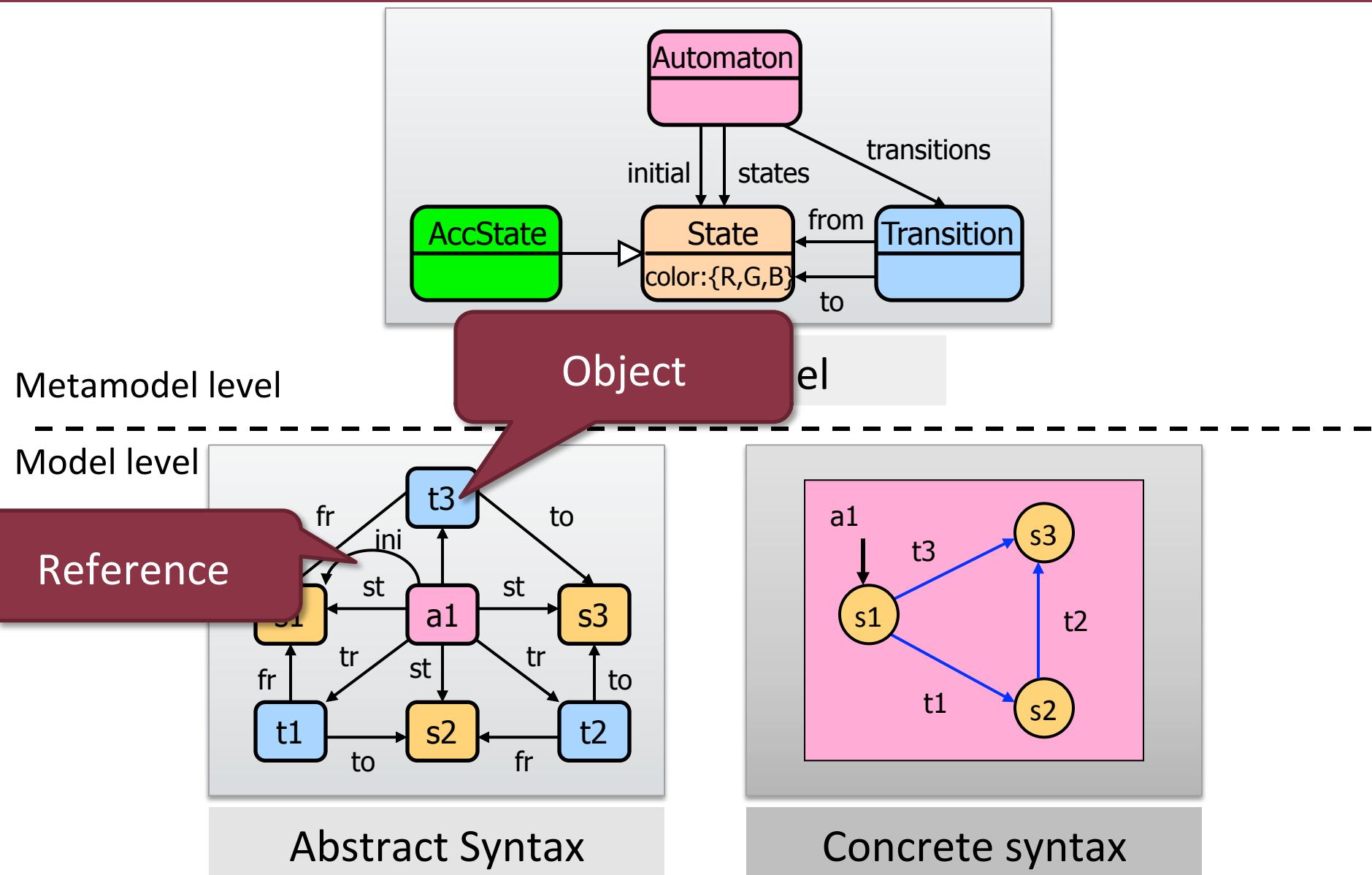


Concrete syntax

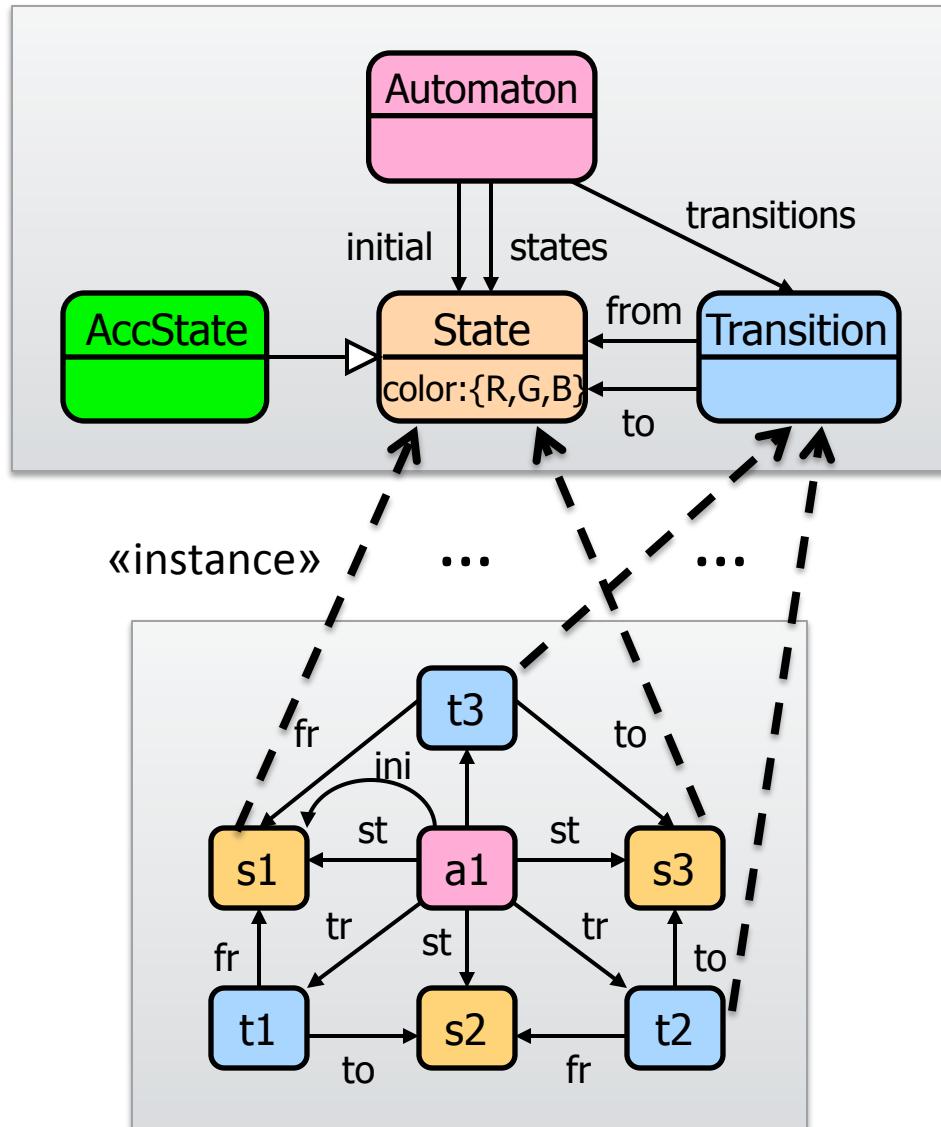
# Meta- and instance models



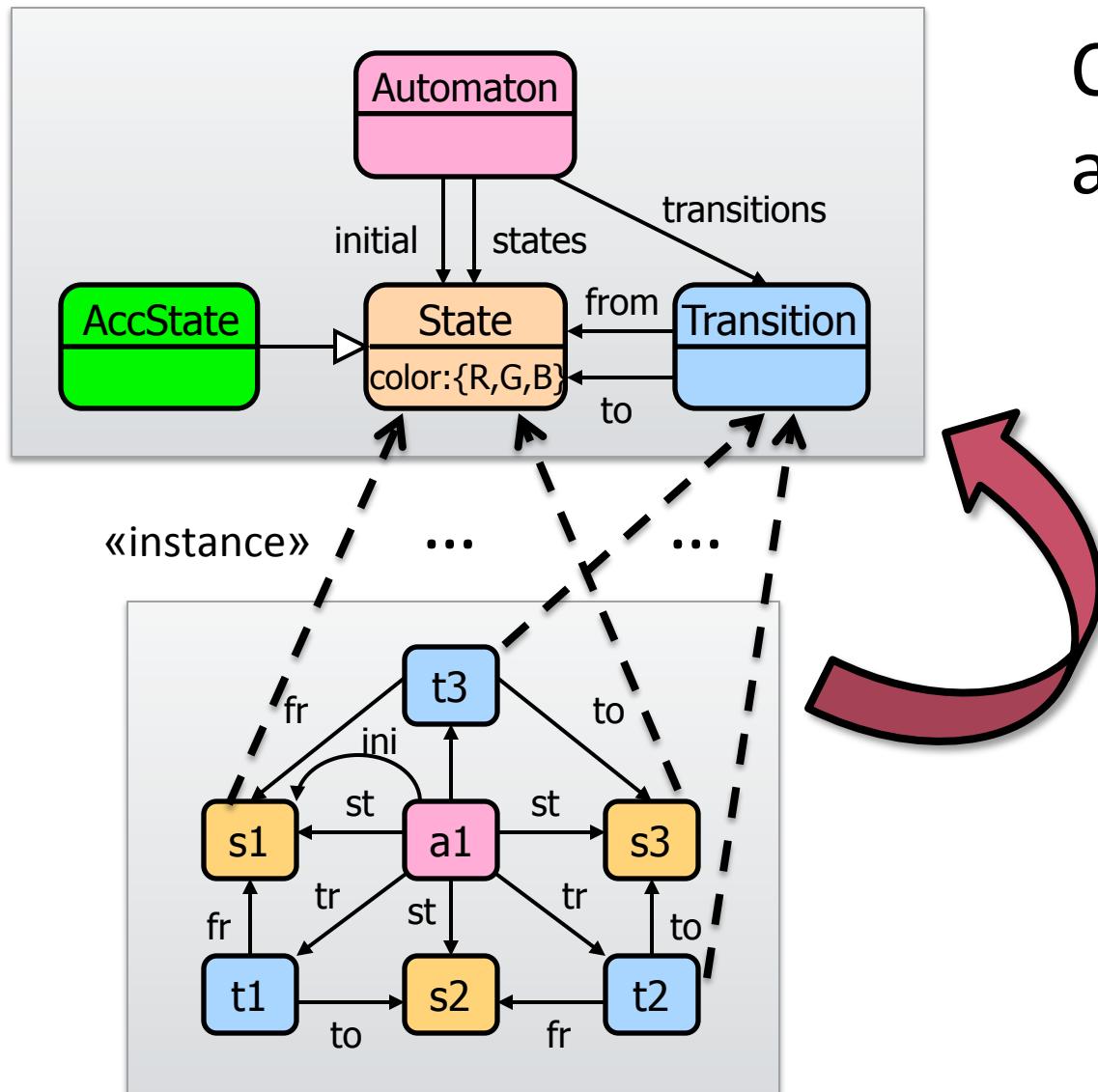
# Meta- and instance models



# Instantiation



# Instantiation



Connects instance  
and metamodel

# Concrete syntax

- A notation for the users
  - As the user “sees” the models
- Multiple concrete syntaxes for every language
  - Either different notations
  - Or different views

# Graphical concrete syntax

- + Easy to read
  - + Intuitive, understandable notation
- + Safe(r) to write
  - + Only syntactically concrete models can be created
- Hard to write
  - Graphical editing is slower
- Scalability issues
  - Usability issues
  - Sometimes technological issues as well
- Related Eclipse technologies: starting next week

# Textual concrete syntax

## + Easy to write

- Even complex expressions can be written quickly

## + Scales well

- Files with 10k lines can be managed

## - Hard to read

- Understandability
- Does not visualize connections

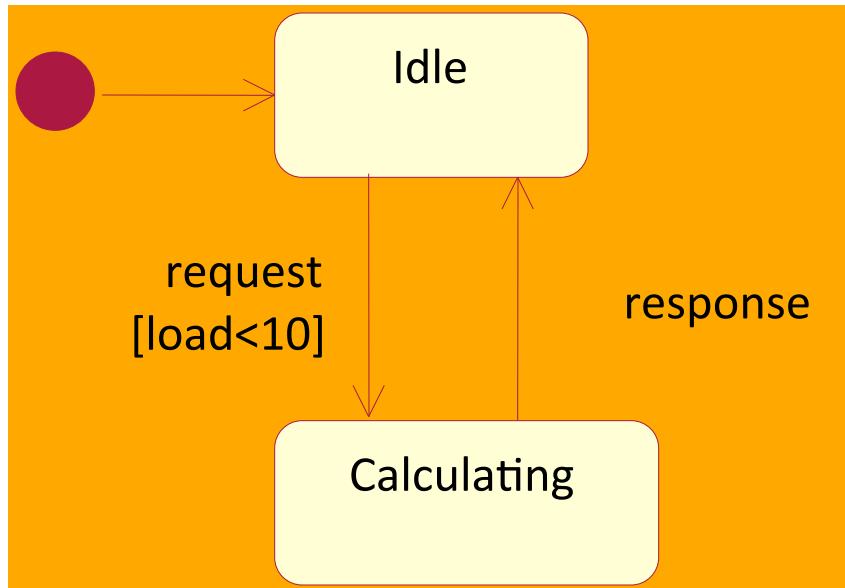
## - Maintenance problematic

- E.g. reference by name

## ■ Corresponding Eclipse technologies: in november

# Example concrete syntax for state machines

- Graphical notation



- Textual notation

```
void request() {  
    if(state == IDLE &&  
        this.load < 10)  
        state =  
CALCULATING;  
}
```

```
void response() {  
    if (state ==  
CALCULATING)  
        state = IDLE;  
}
```

# Abstract syntax

- Defines the vocabulary of the languages
  - Terms
  - Relations between terms
    - Associations/references
    - Attributes
    - Abstractions/refinements (see taxonomies, ontologies)

# Well-formedness rules

- Multiplicity constraints
  - One: 1
  - At most one: 0..1
  - Many: \*
- Aggregation/Containment references
  - Specifies a containment hierarchy
  - At most one parent for each model element
- Language-specific constraints
  - E.g., unique names

# Dynamic semantics

- Semantics:
  - Meaning of the terms of the language
  - How to understand the models
- Static semantics
  - What are the valid models?
  - See previous slides
- Dynamic semantics
  - Possible behaviour
  - State changes

# Dynamic semantics – Main approaches

- Denotational semantics
  - Translating terms to another language
  - Similar to compilers
- Operation semantics
  - Modeling the behavior of terms
  - Similar to interpreters
- ~~Axiomatic semantics~~
  - ~~Based on logical formulae~~
  - ~~Hard to understand/implement~~

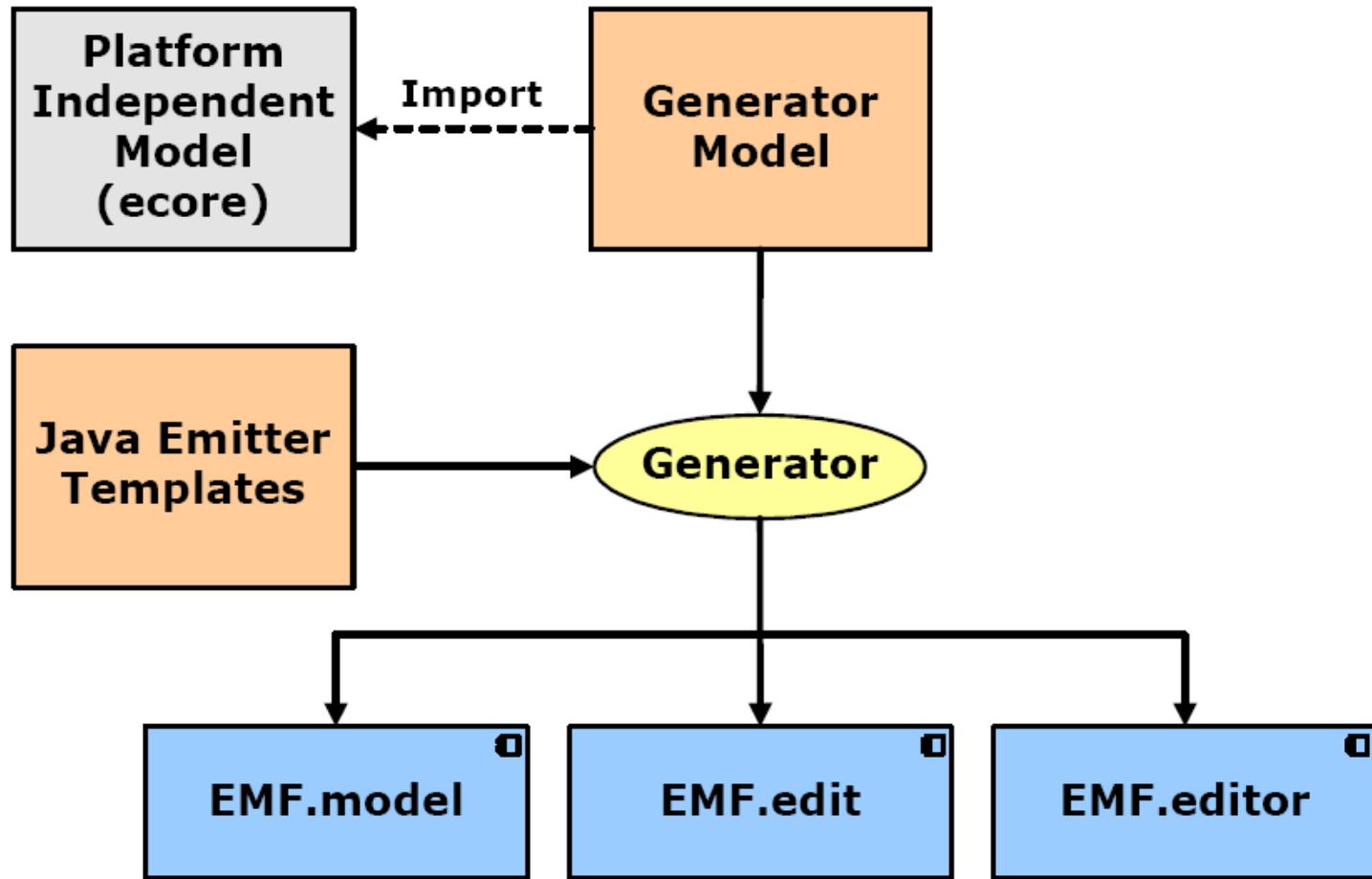
# Eclipse Modeling Framework (EMF)

Metamodeling in Eclipse

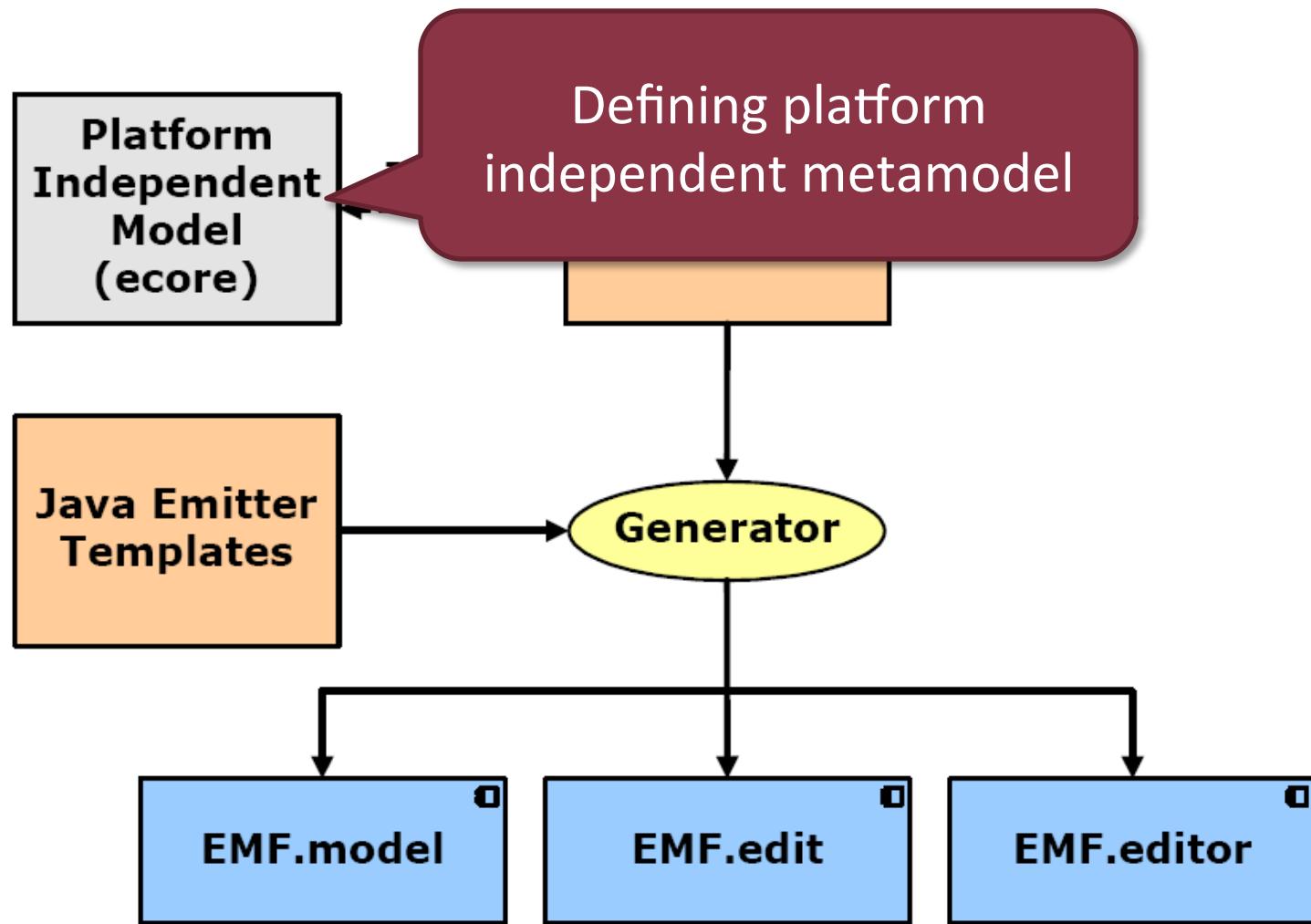
# Eclipse Modeling Framework

- Modeling component for Eclipse
  - Supports the definition of DSLs
- Supports
  - Basic editing commands
  - Change notification
  - Undo/redo support
  - XML/XMI export-import
- Uses a simple metamodeling core
  - Called Ecore
  - Similar to MOF (used for UML)

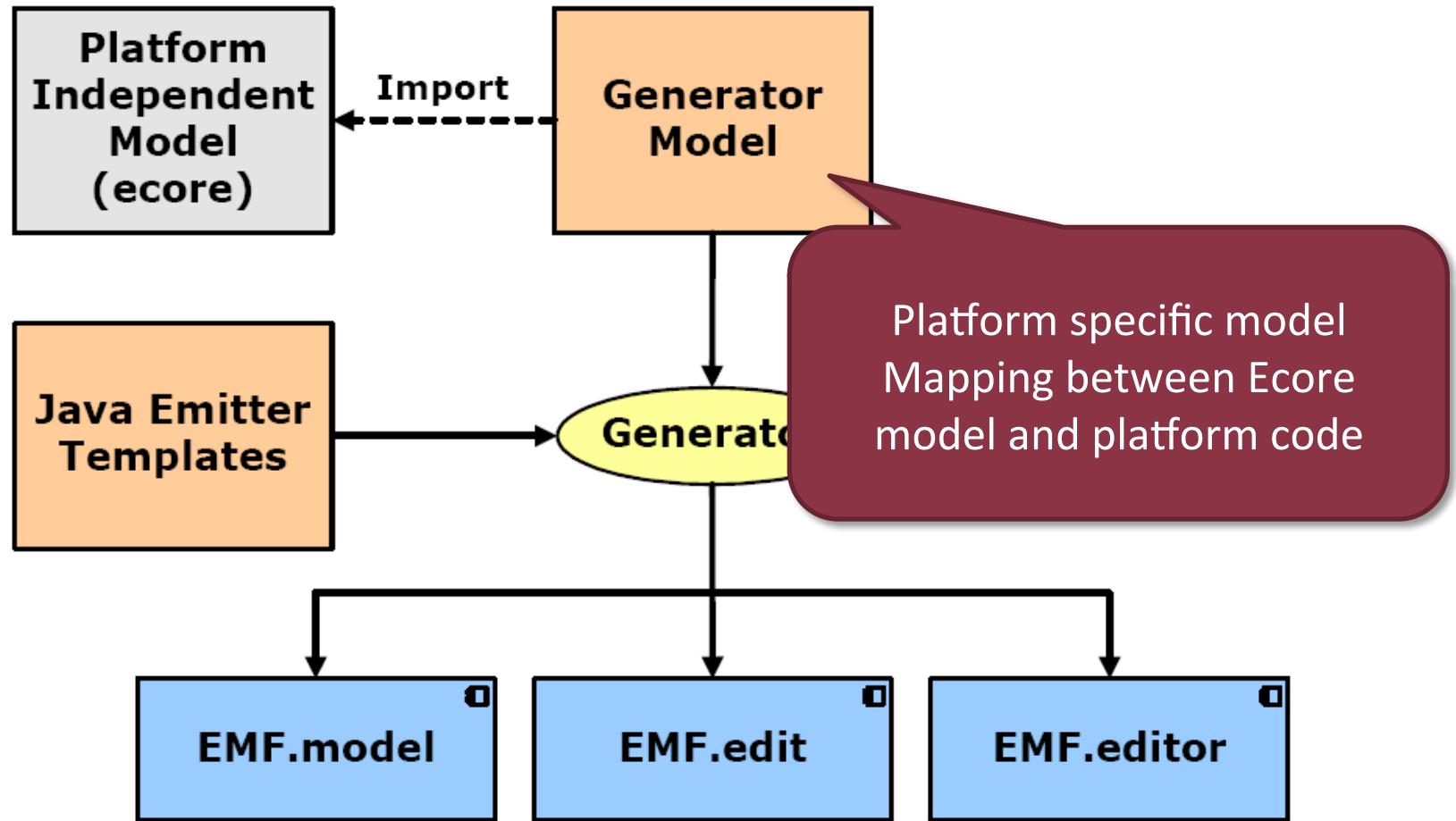
# The EMF tooling



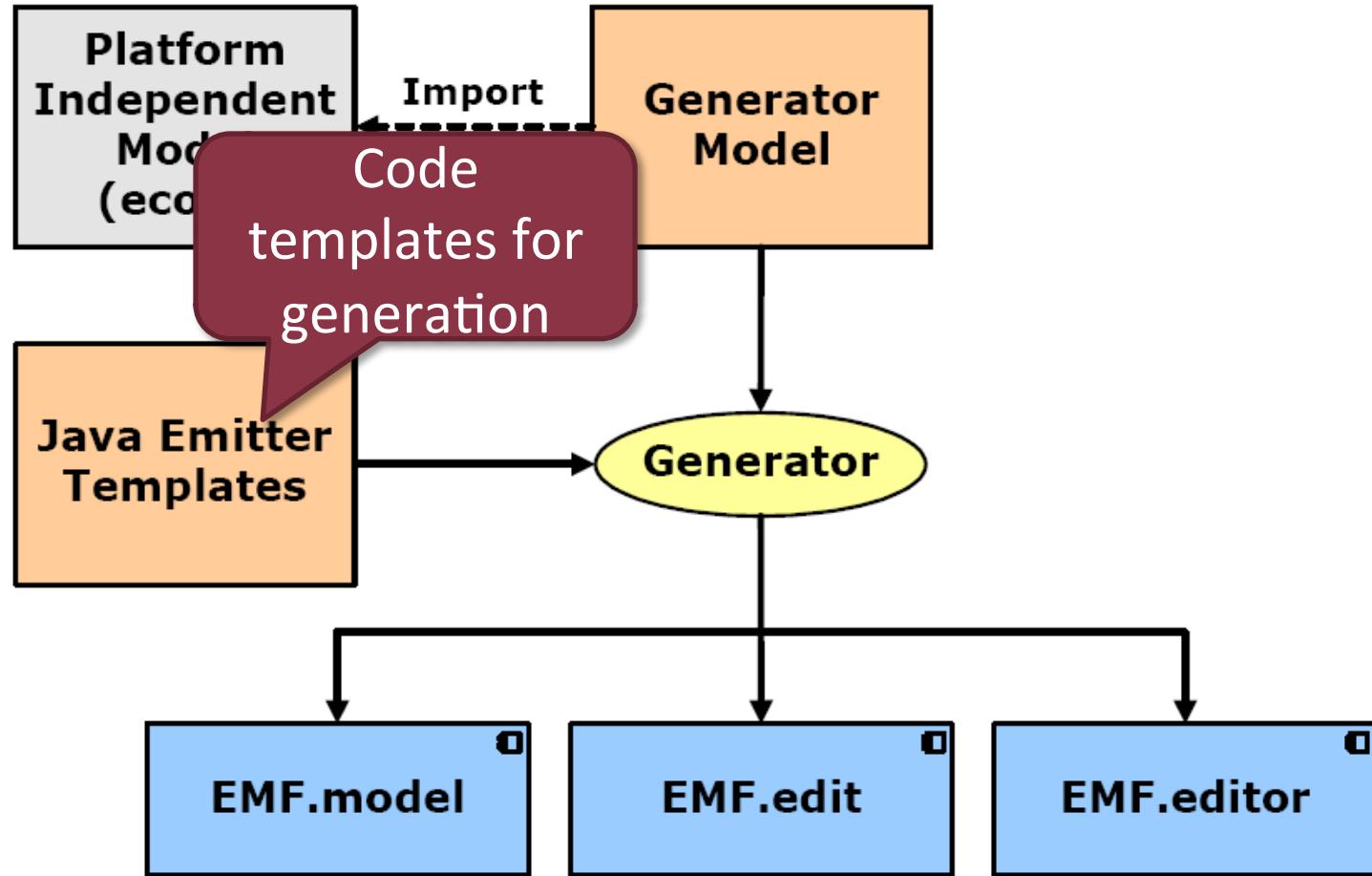
# The EMF tooling



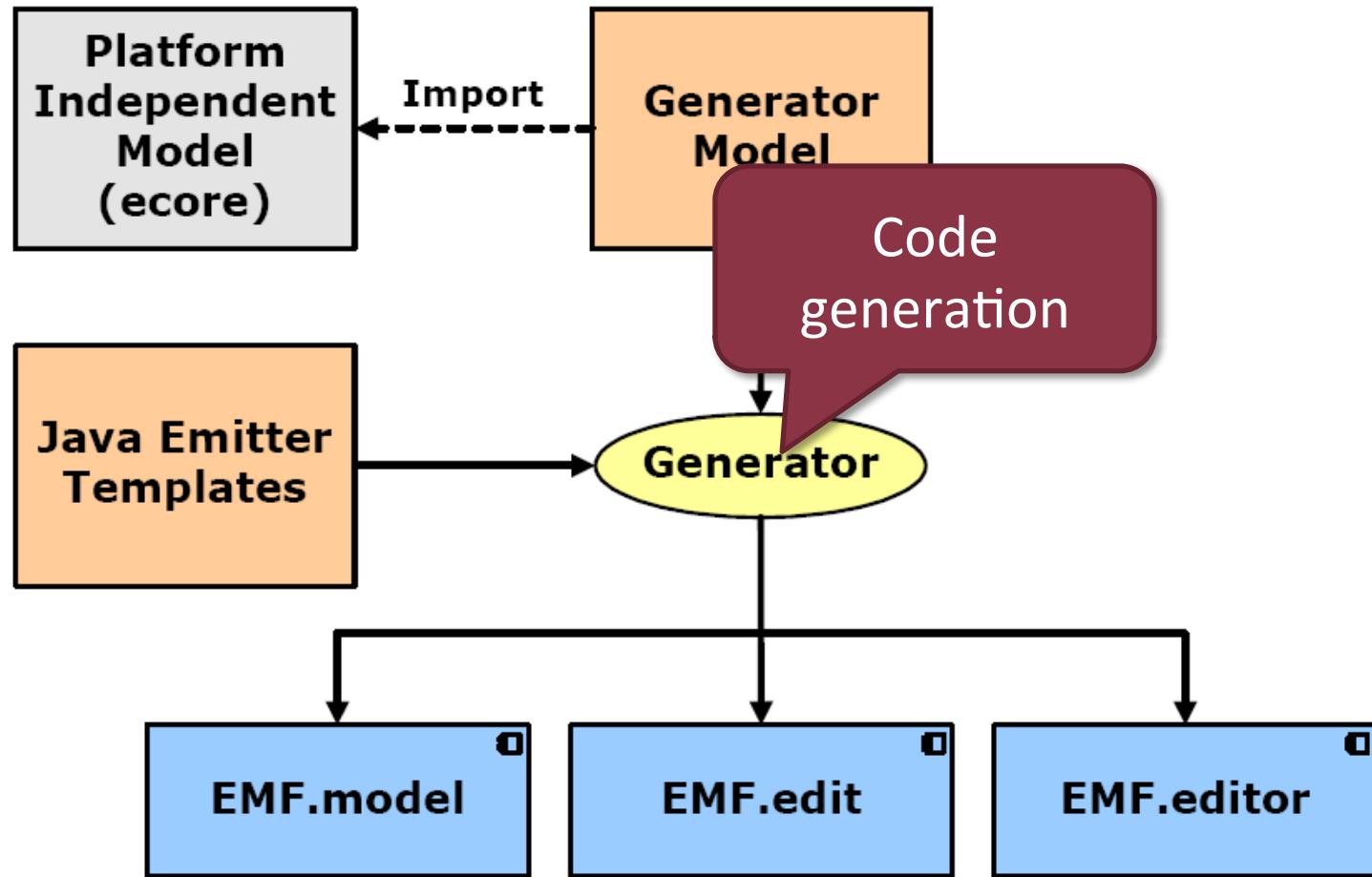
# The EMF tooling



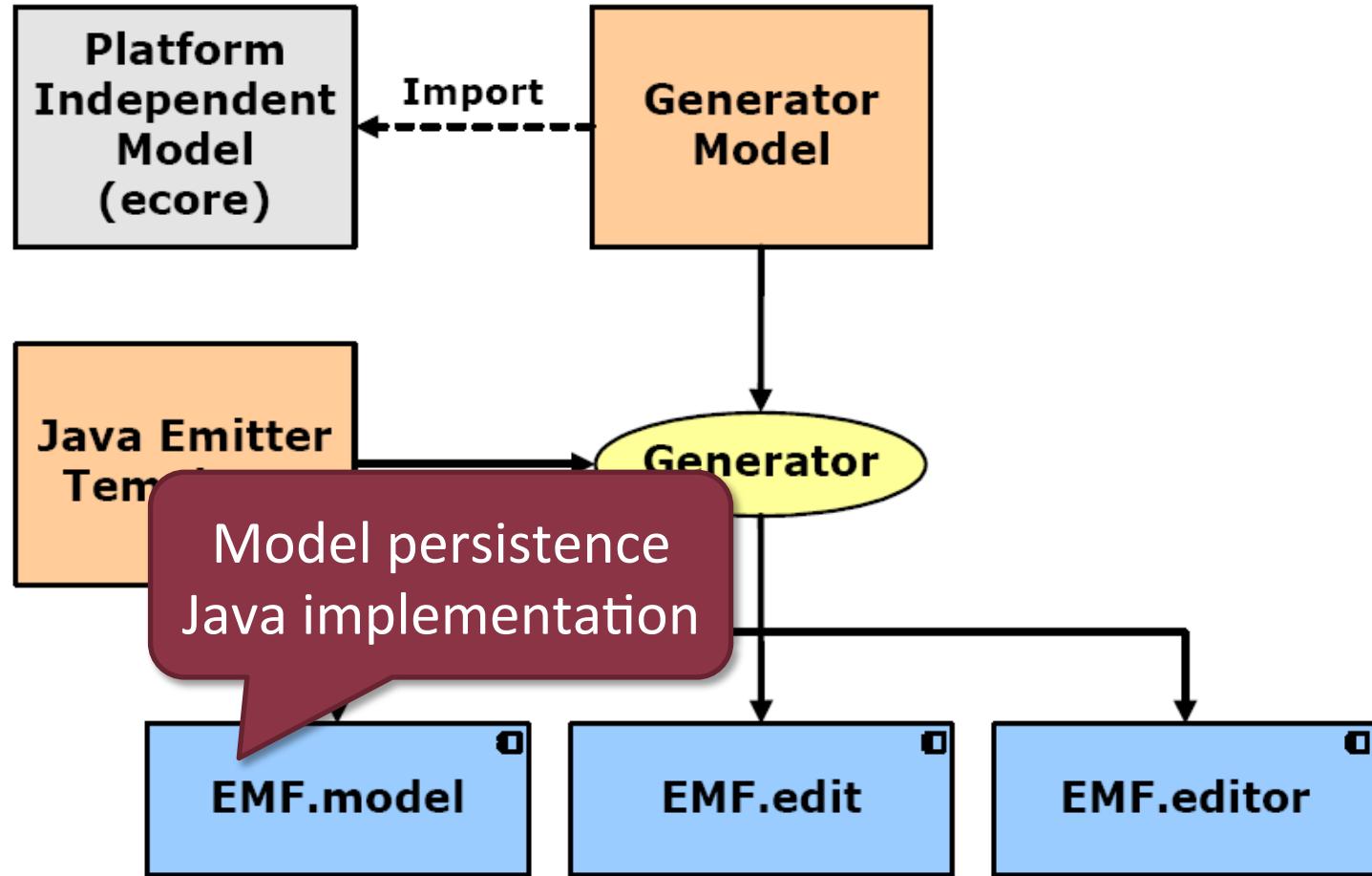
# The EMF tooling



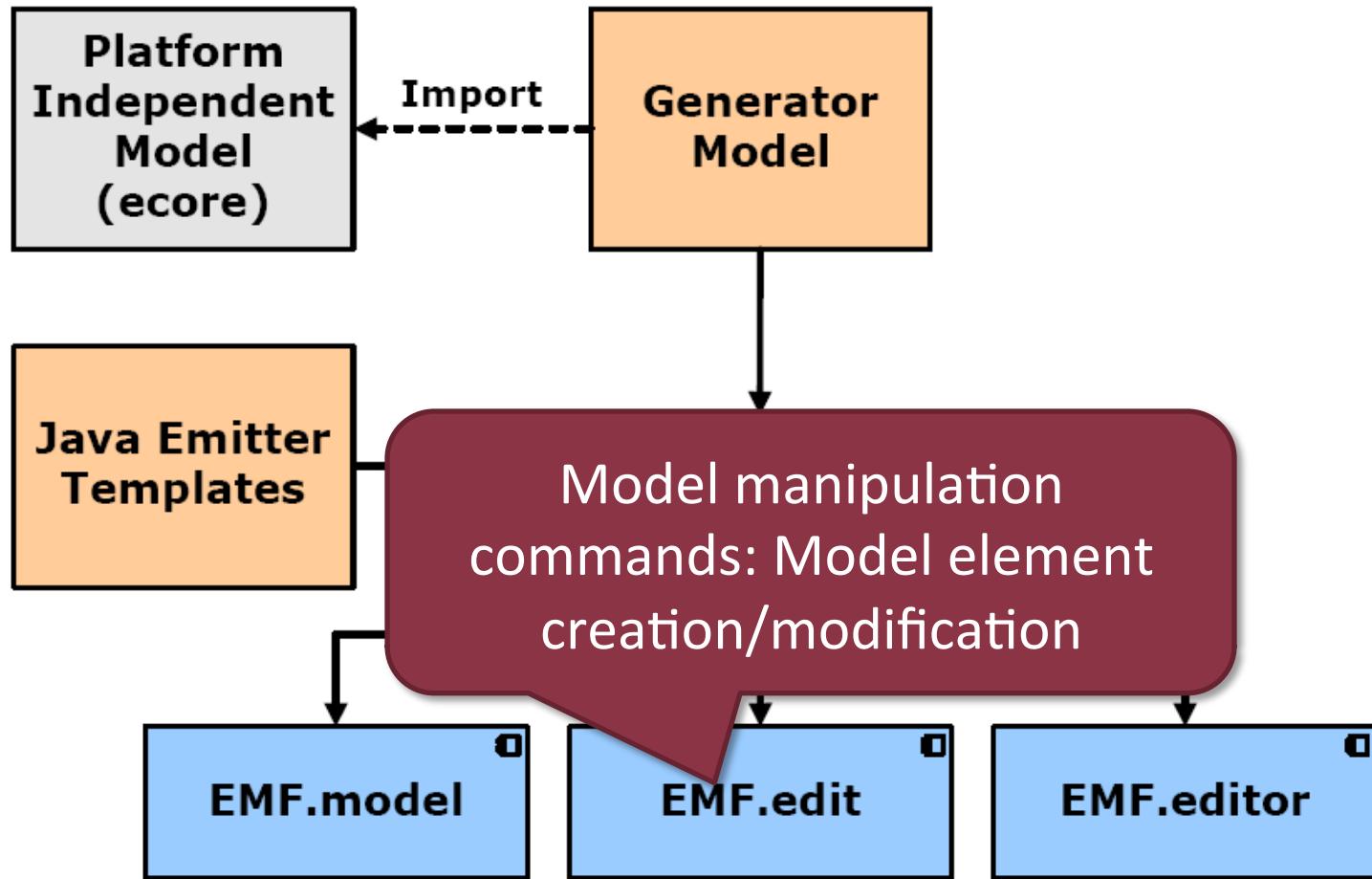
# The EMF tooling



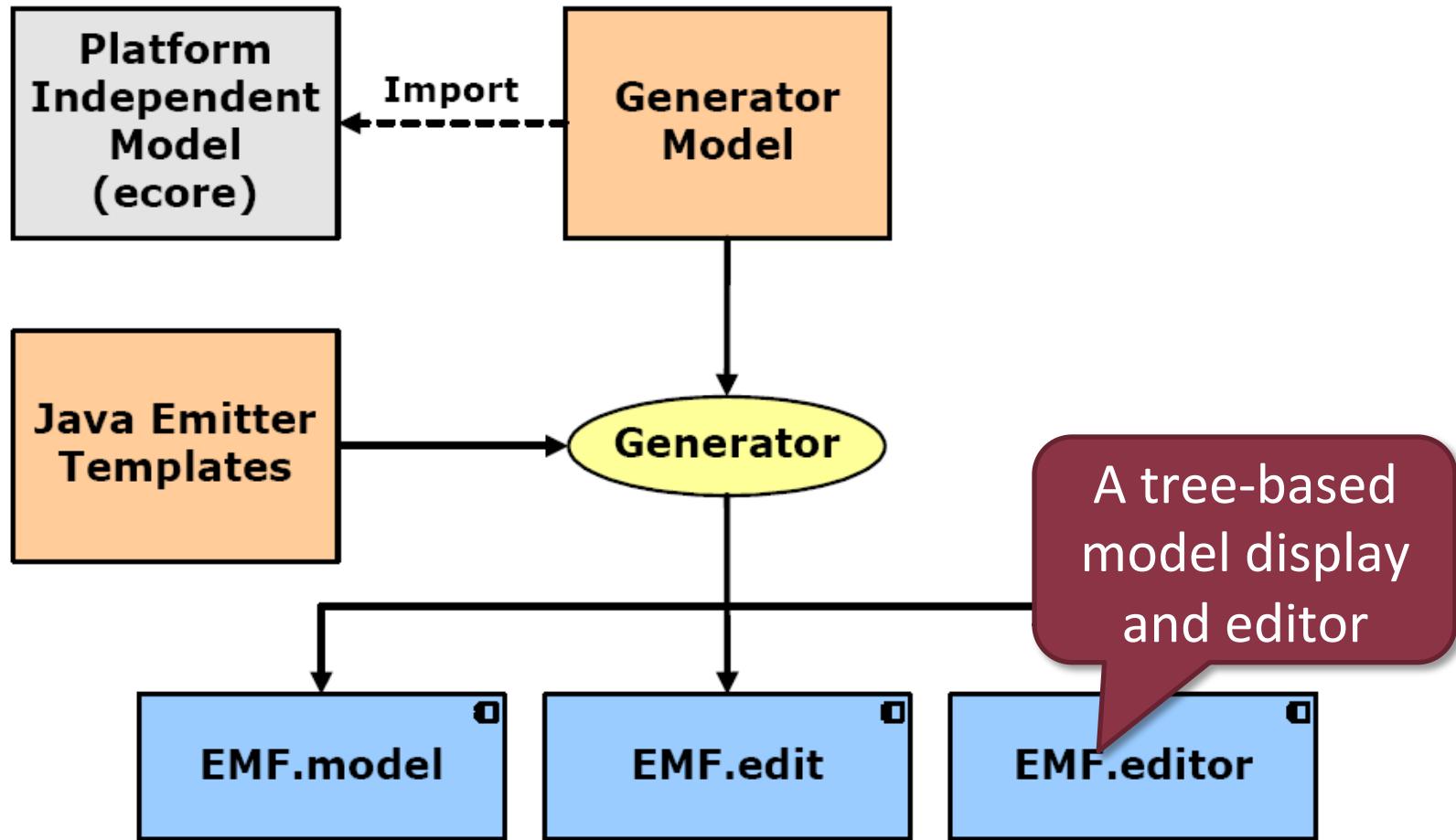
# The EMF tooling



# The EMF tooling



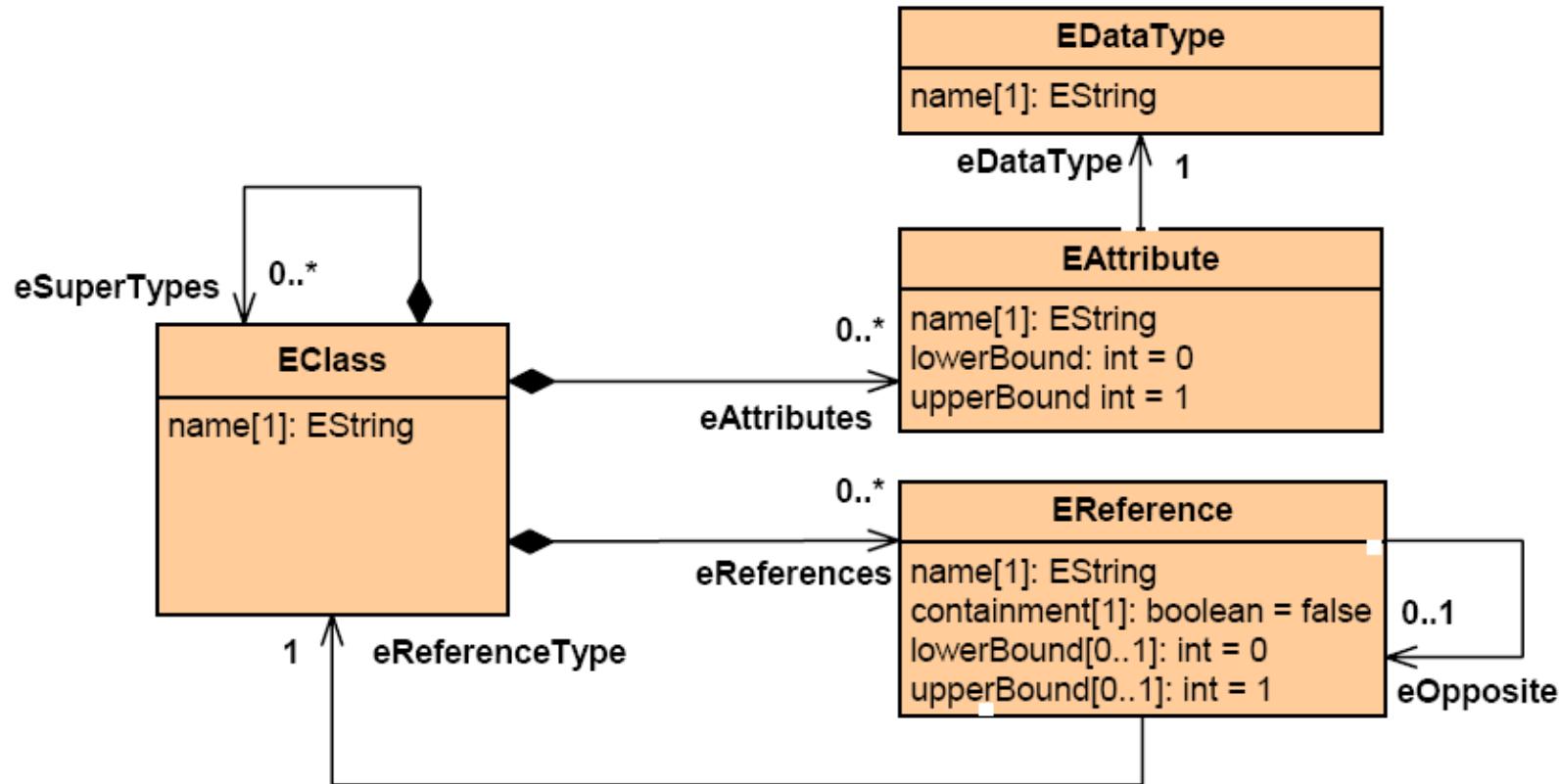
# The EMF tooling



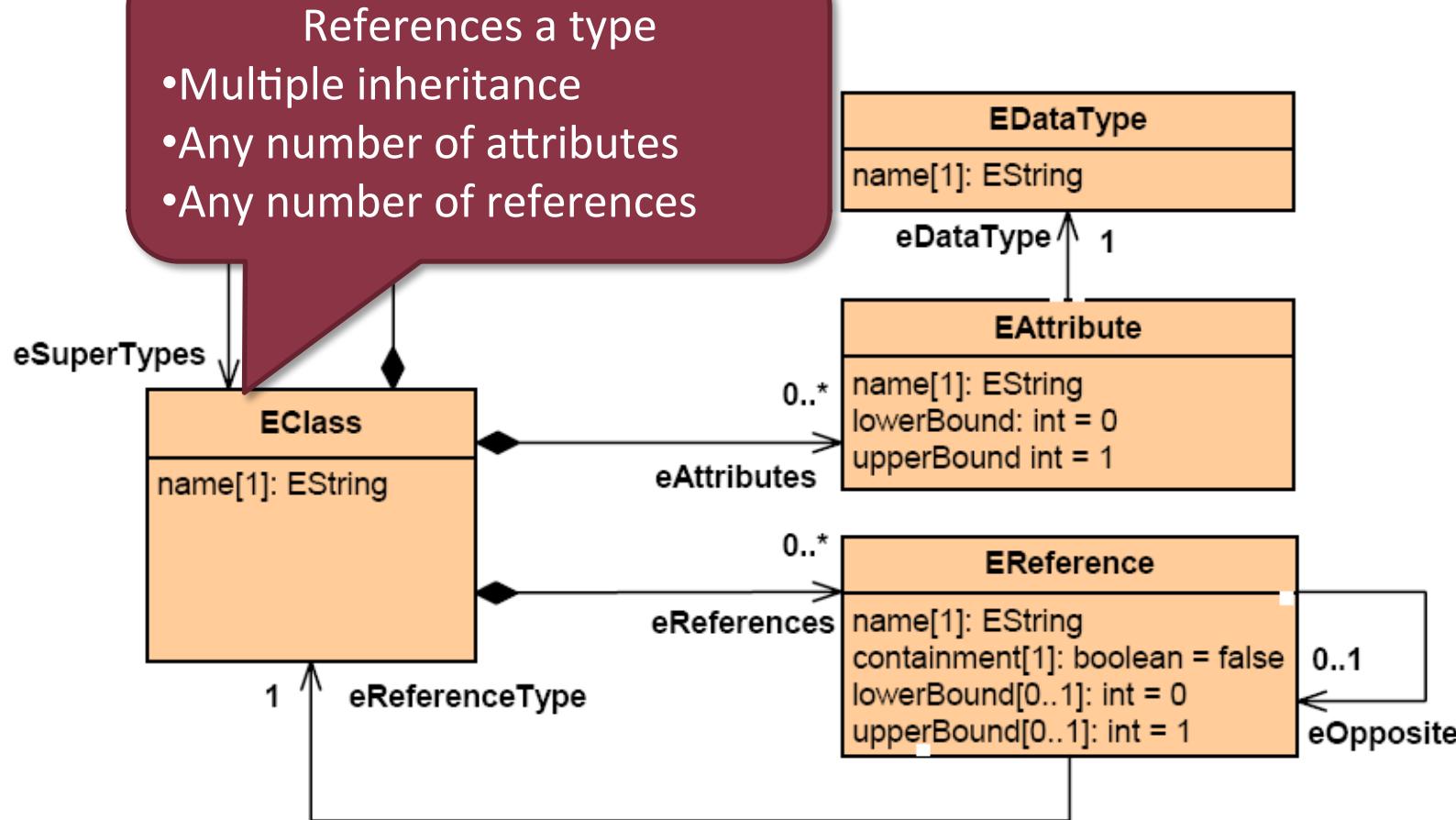
# Ecore language

- Metamodeling language of EMF
  - Meta-language
- Platform independent metamodels
  - Additional models for platform-specific modeling (see genmodel)
- Defines structural information

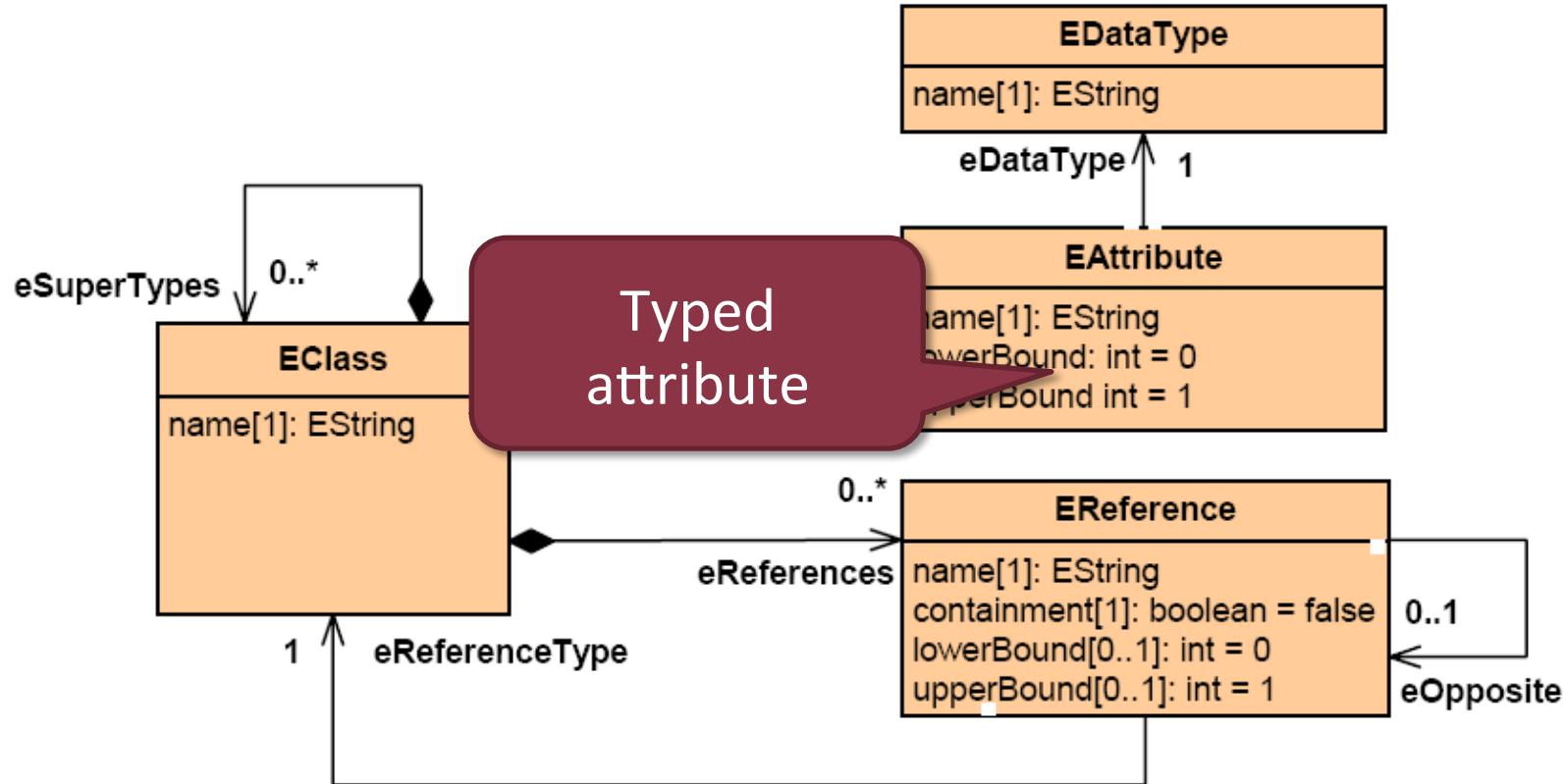
# Ecore – Most important concepts



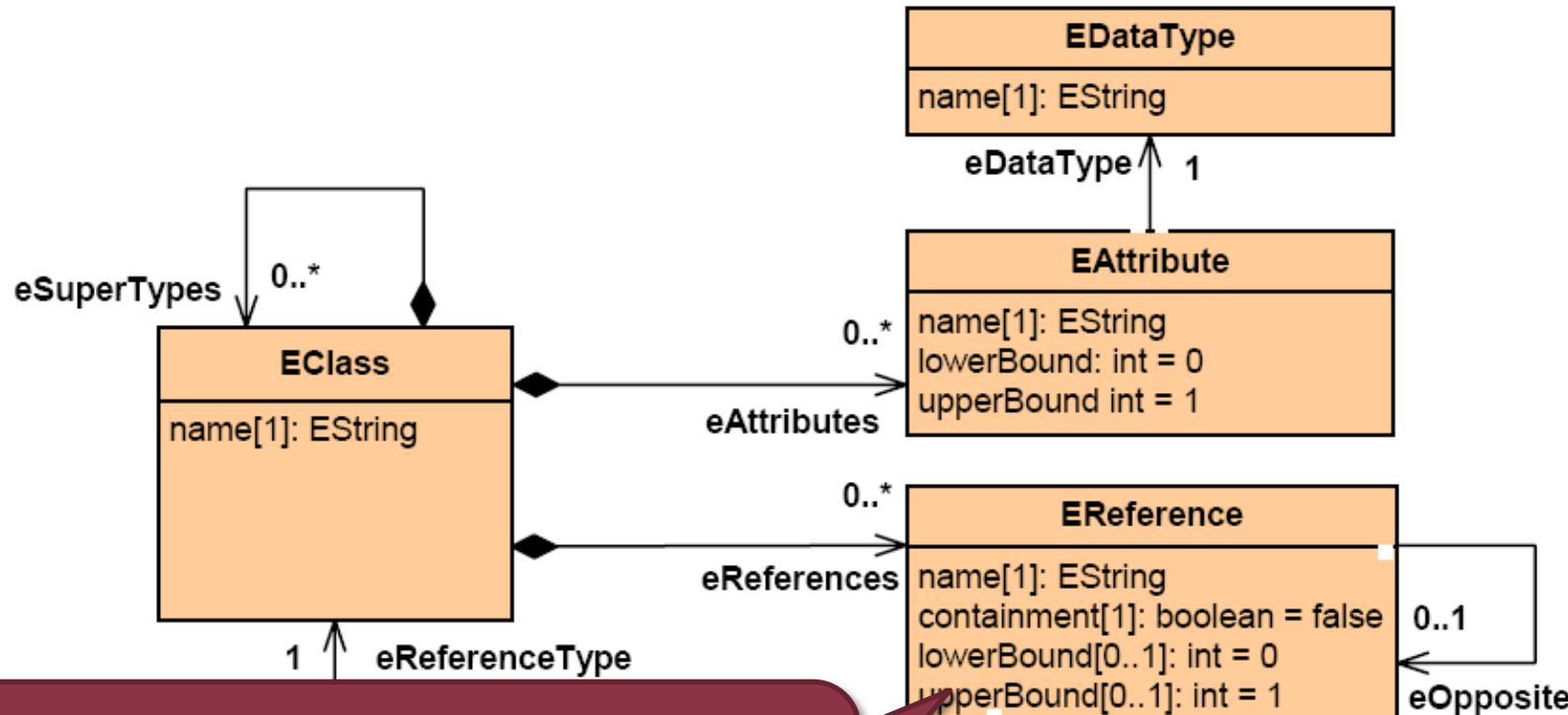
# Ecore – Most important concepts



# Ecore – Most important concepts



# Ecore – Most important concepts



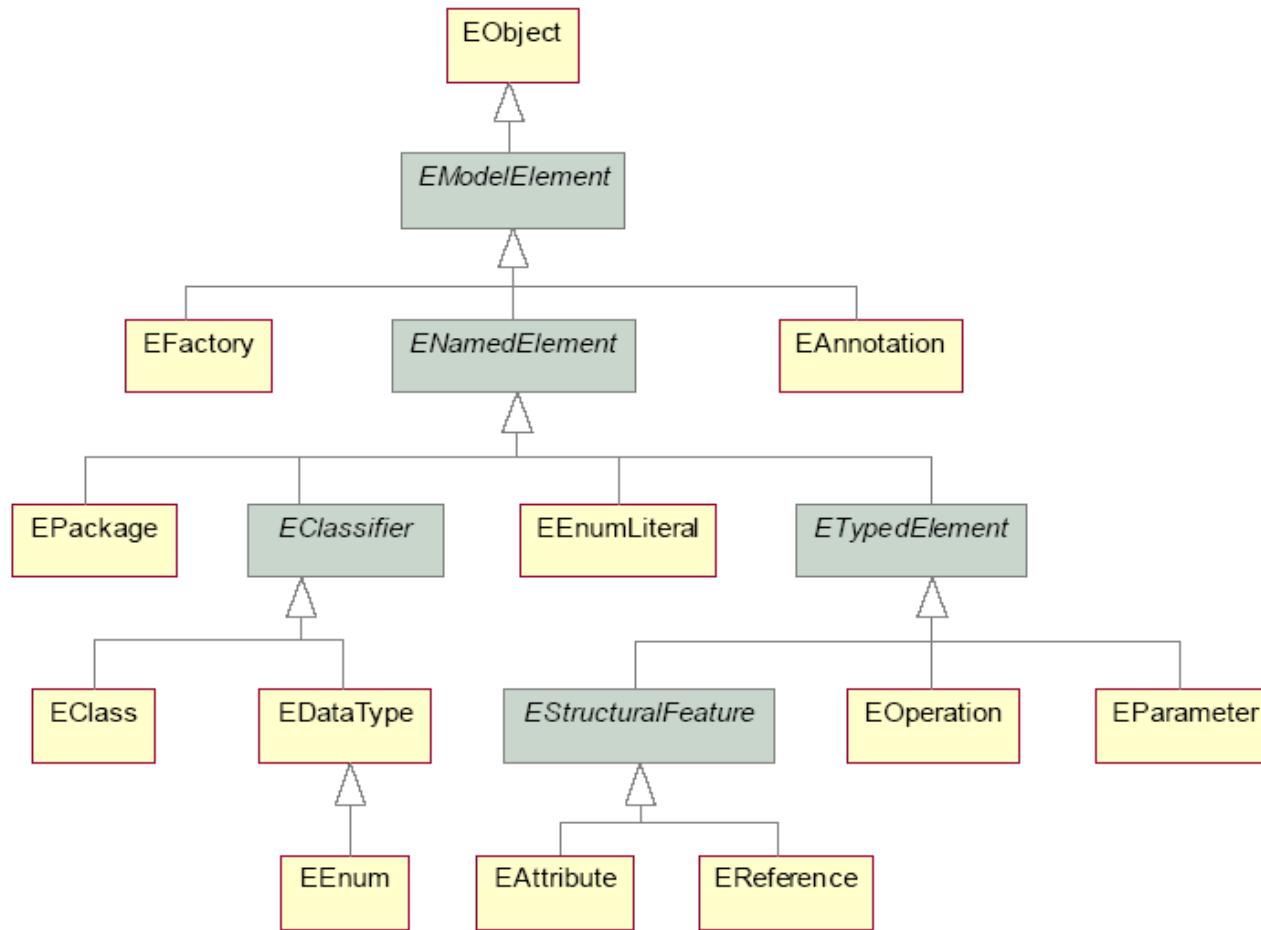
Unidirectional, binary connection  
• Optionally inverse reference  
• Defines referenced type

# Ecore – Most important concepts

## ■ EPackage

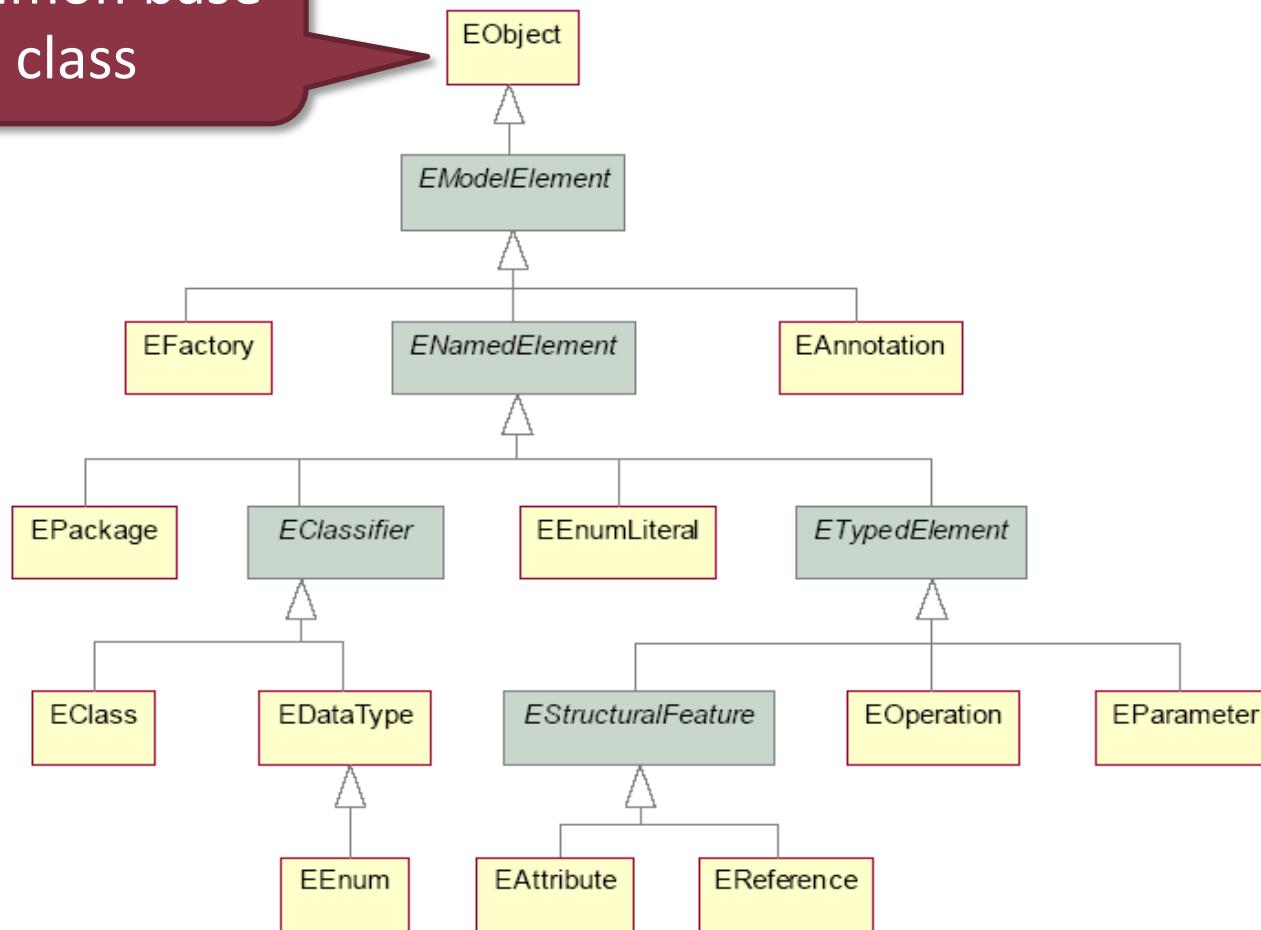
- Contains and manages a set of classes
- Compile/code generation time
  - Builds together
- In runtime
  - Are registered together
  - Provides API for
    - Factory methods
    - Reflective access

# Full Ecore hierarchy

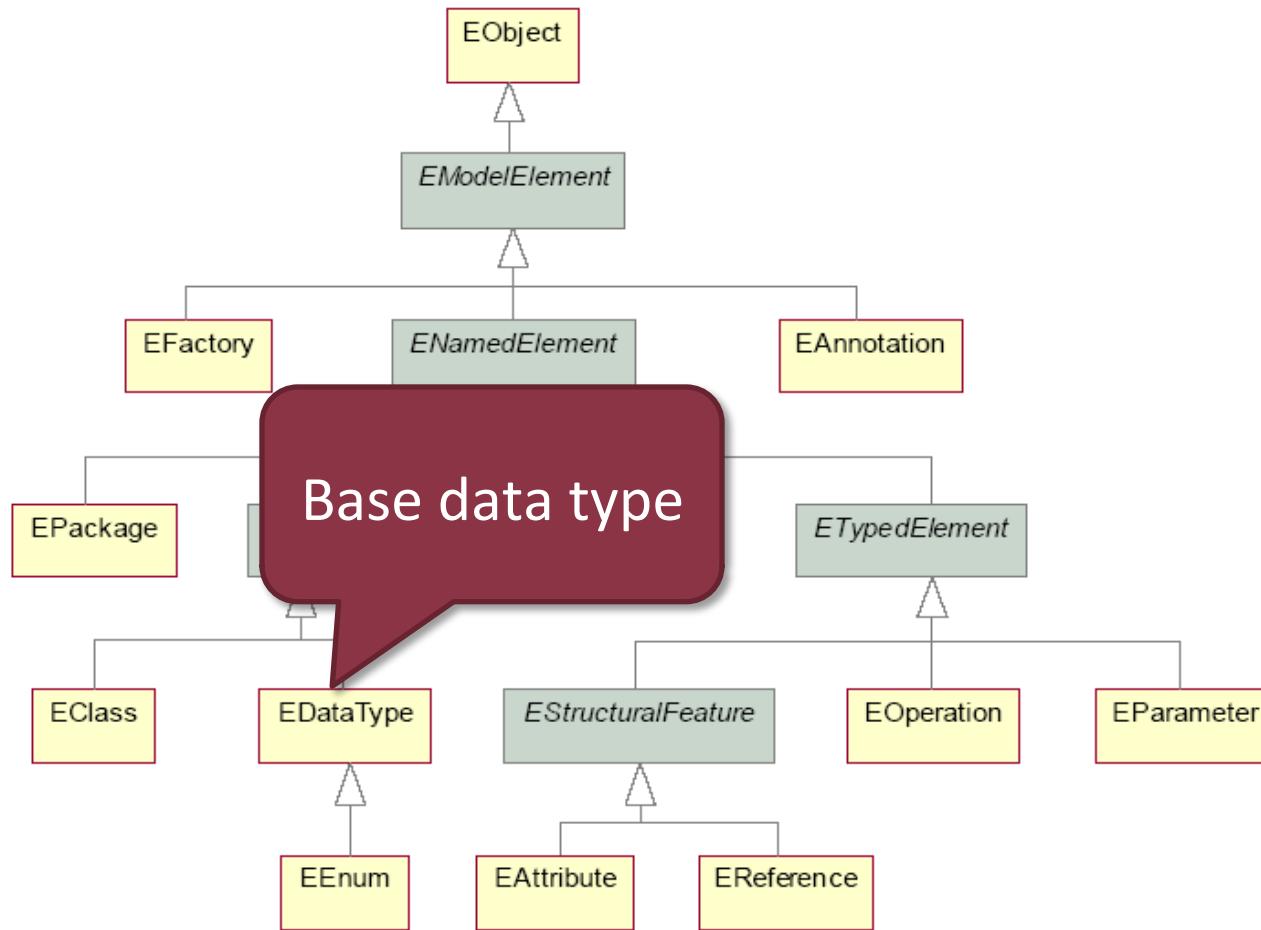


# Full Ecore hierarchy

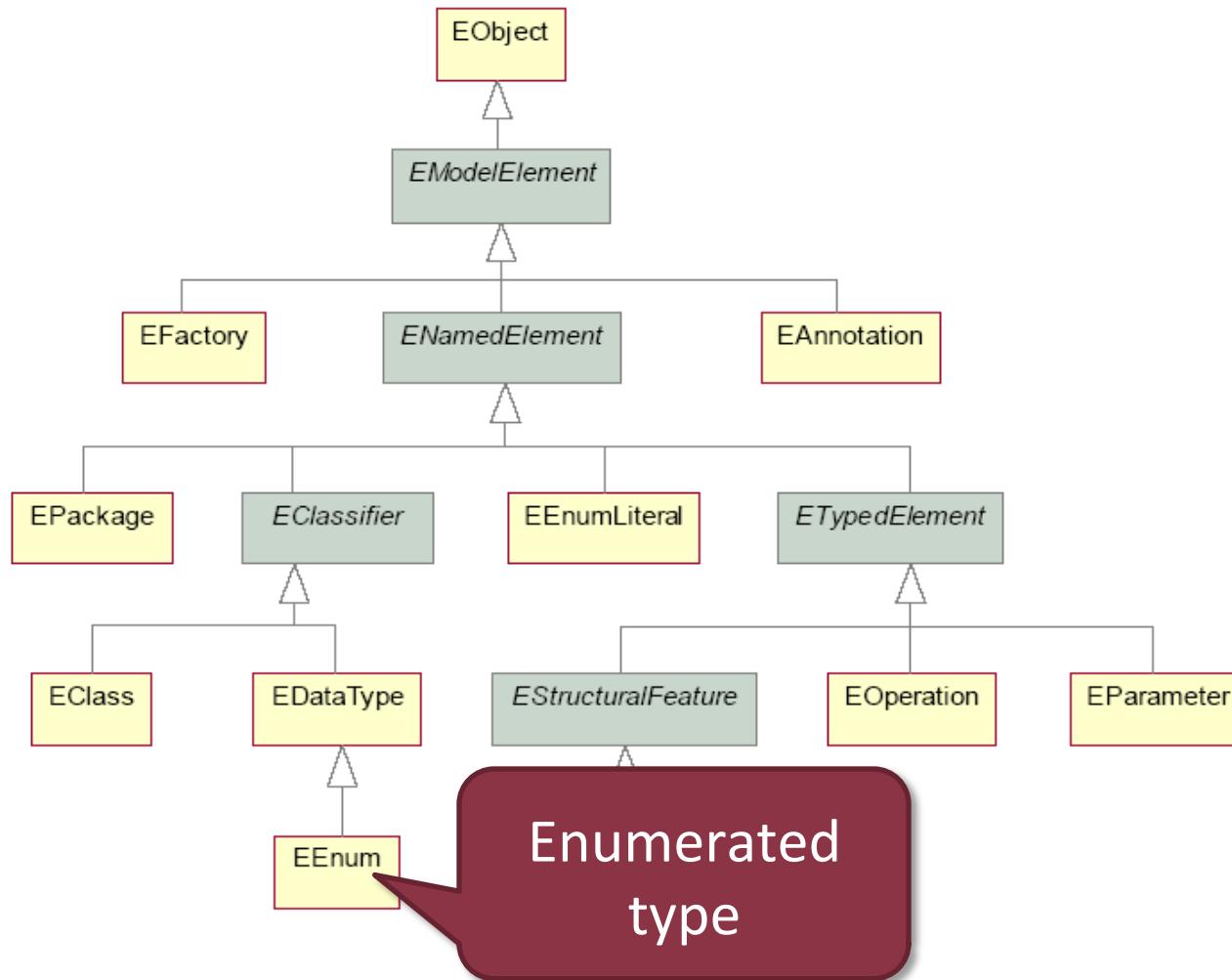
Common base  
class



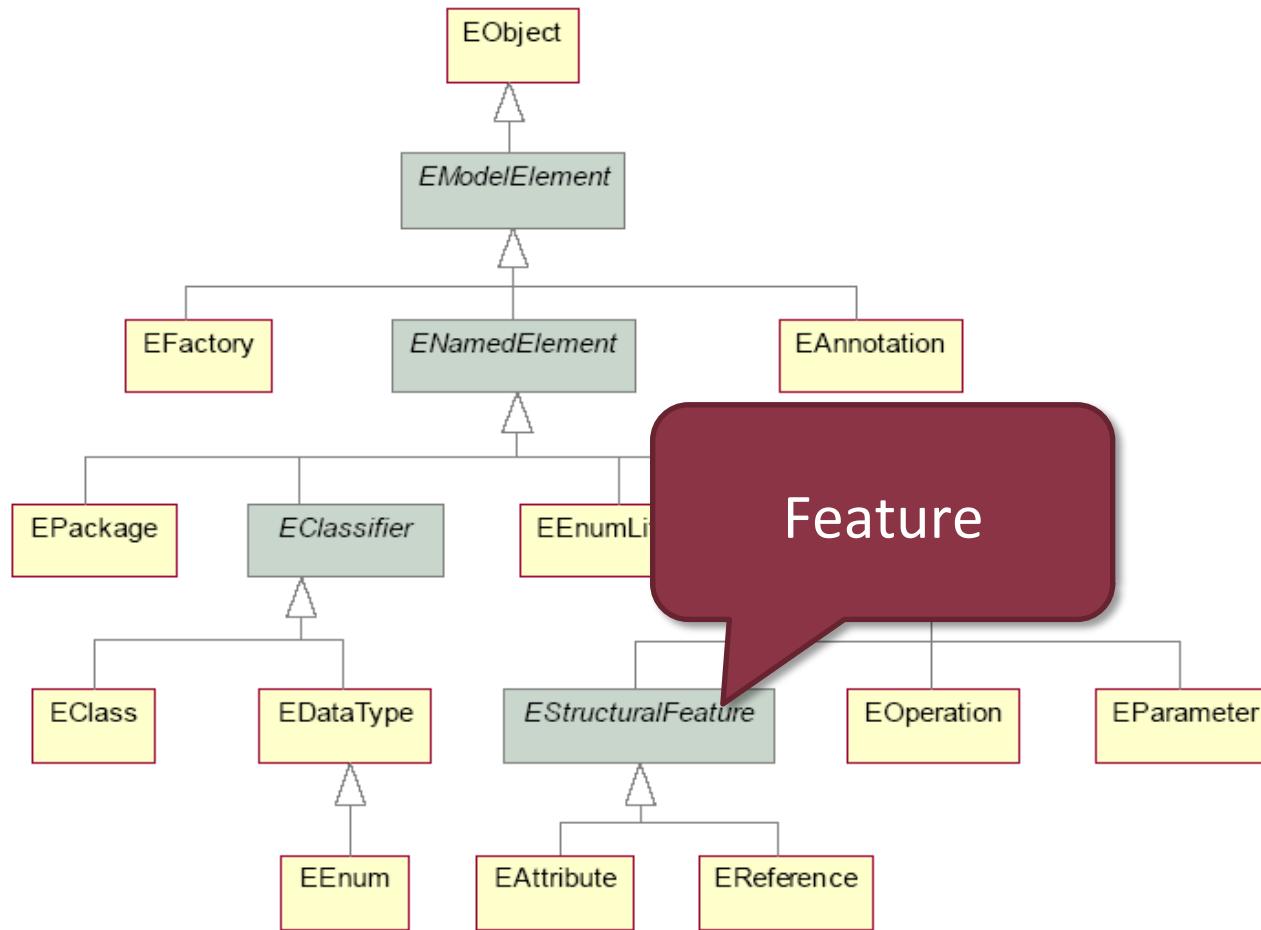
# Full Ecore hierarchy



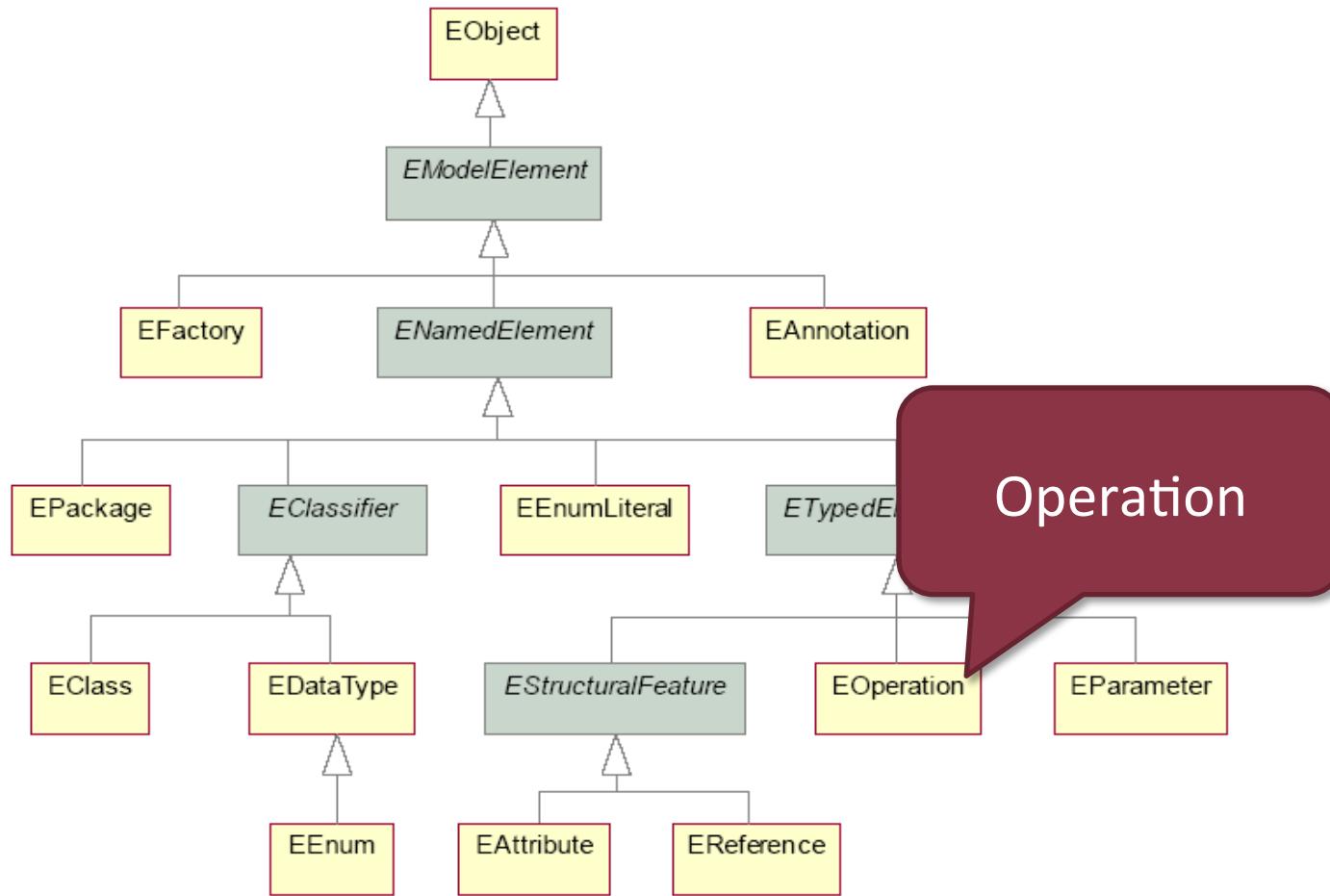
# Full Ecore hierarchy



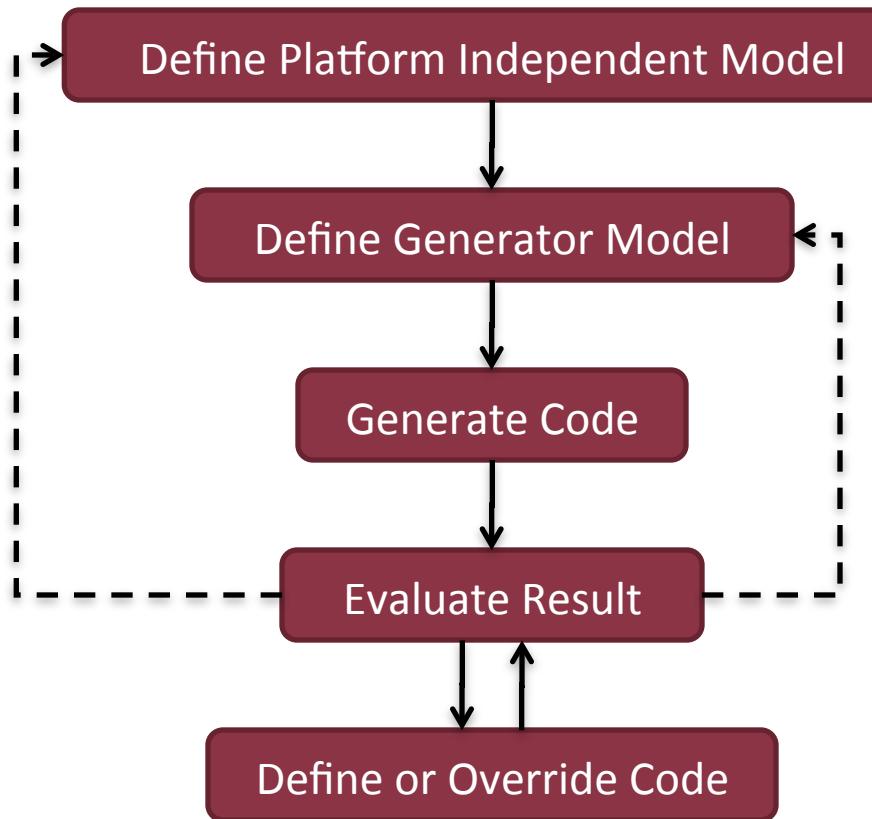
# Full Ecore hierarchy



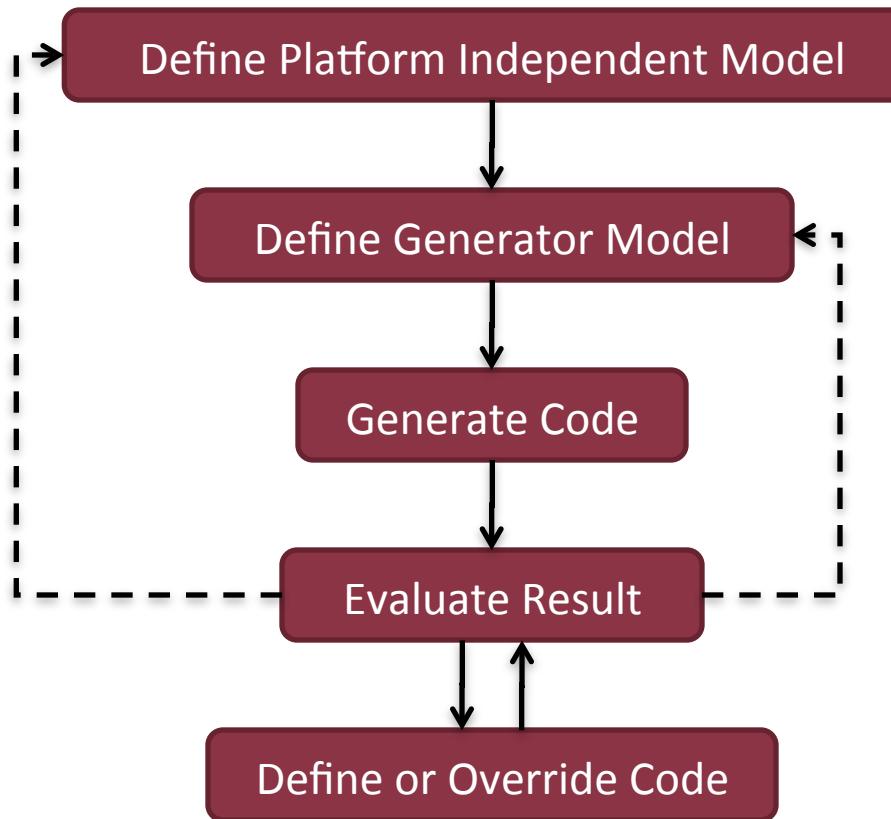
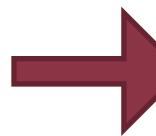
# Full Ecore hierarchy



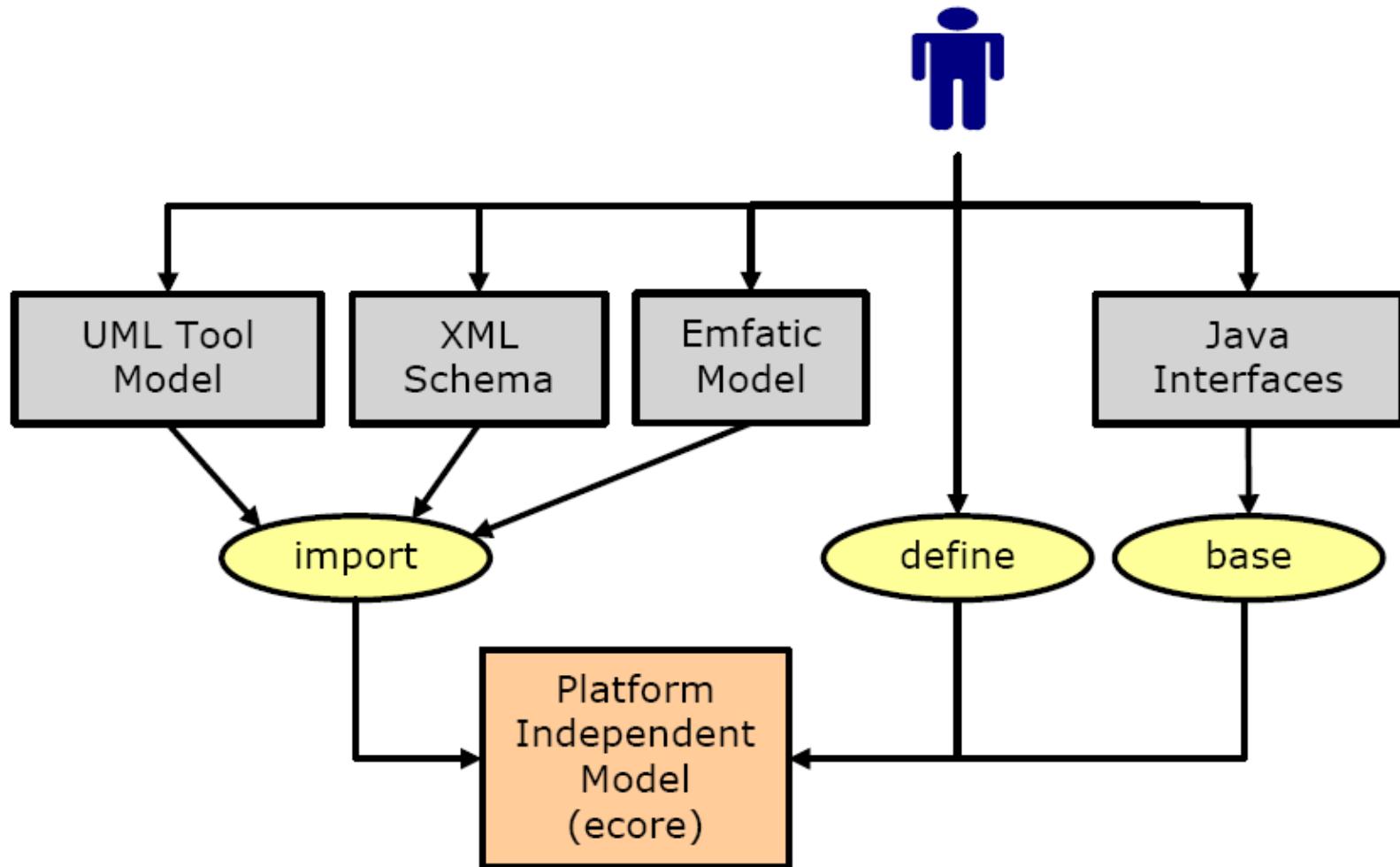
# Using EMF



# Using EMF

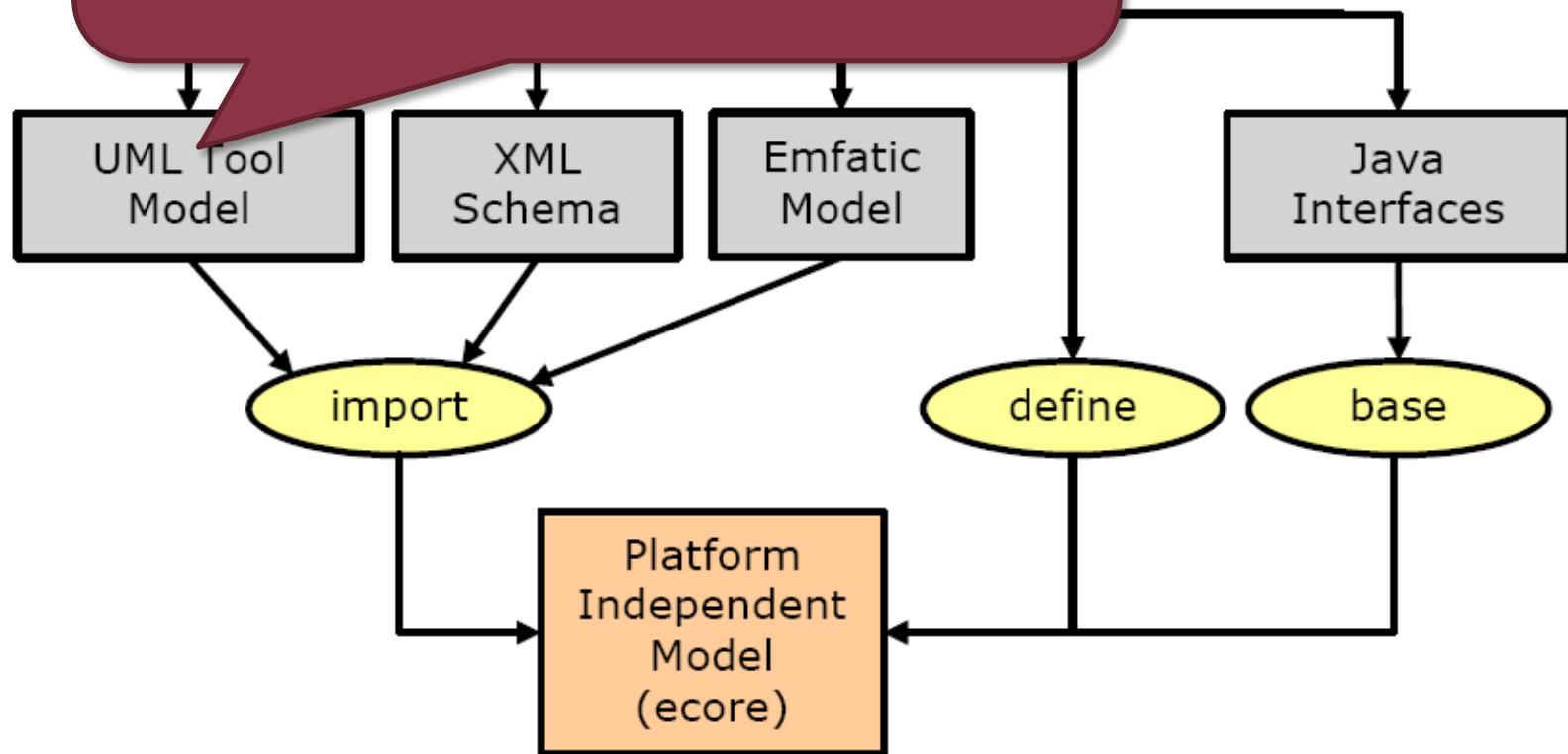


# Defining Ecore Models

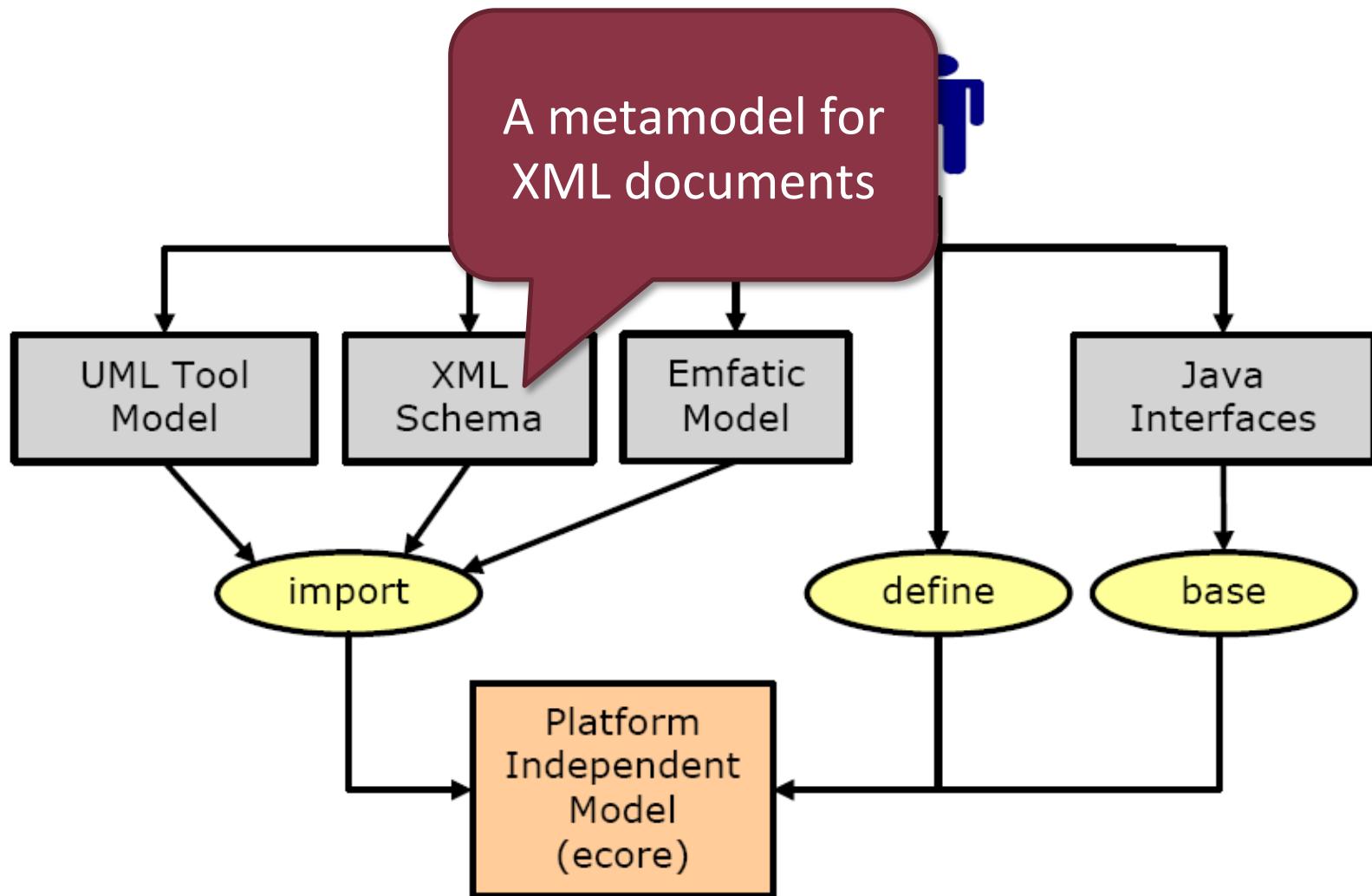


# Defining Ecore Models

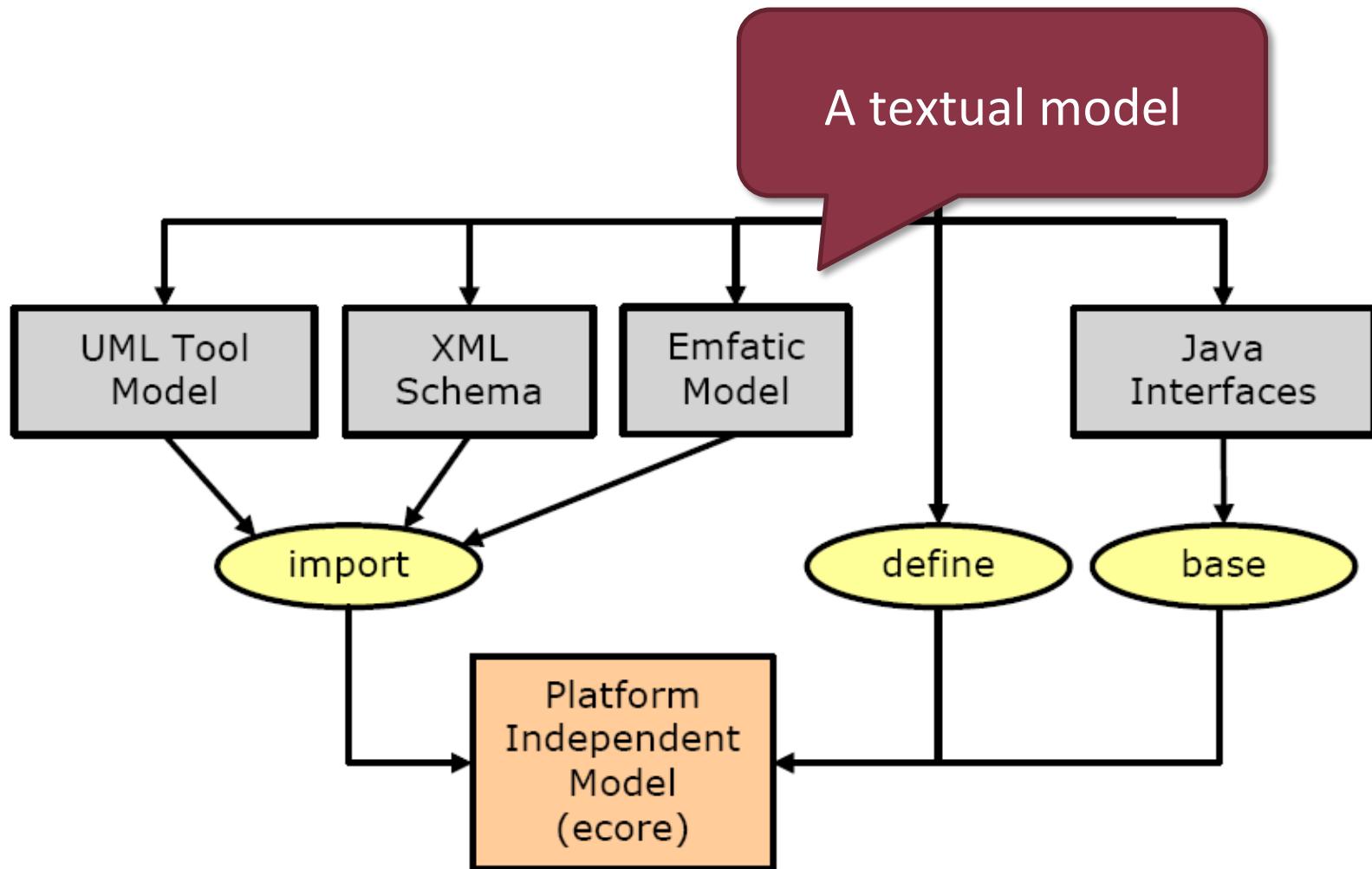
Serialized UML class diagram  
E.g. Rational Software Architect  
EclipseUML



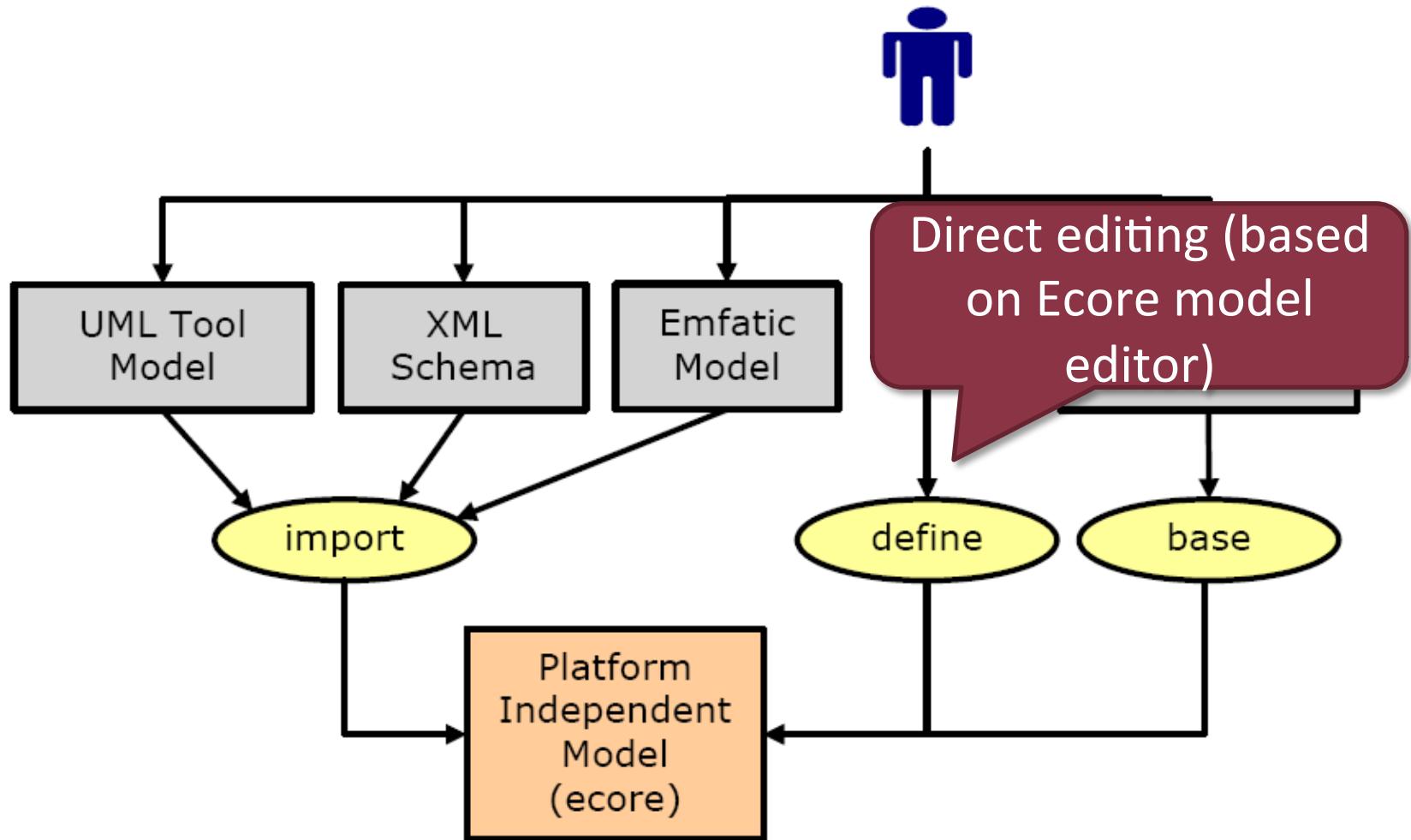
# Defining Ecore Models



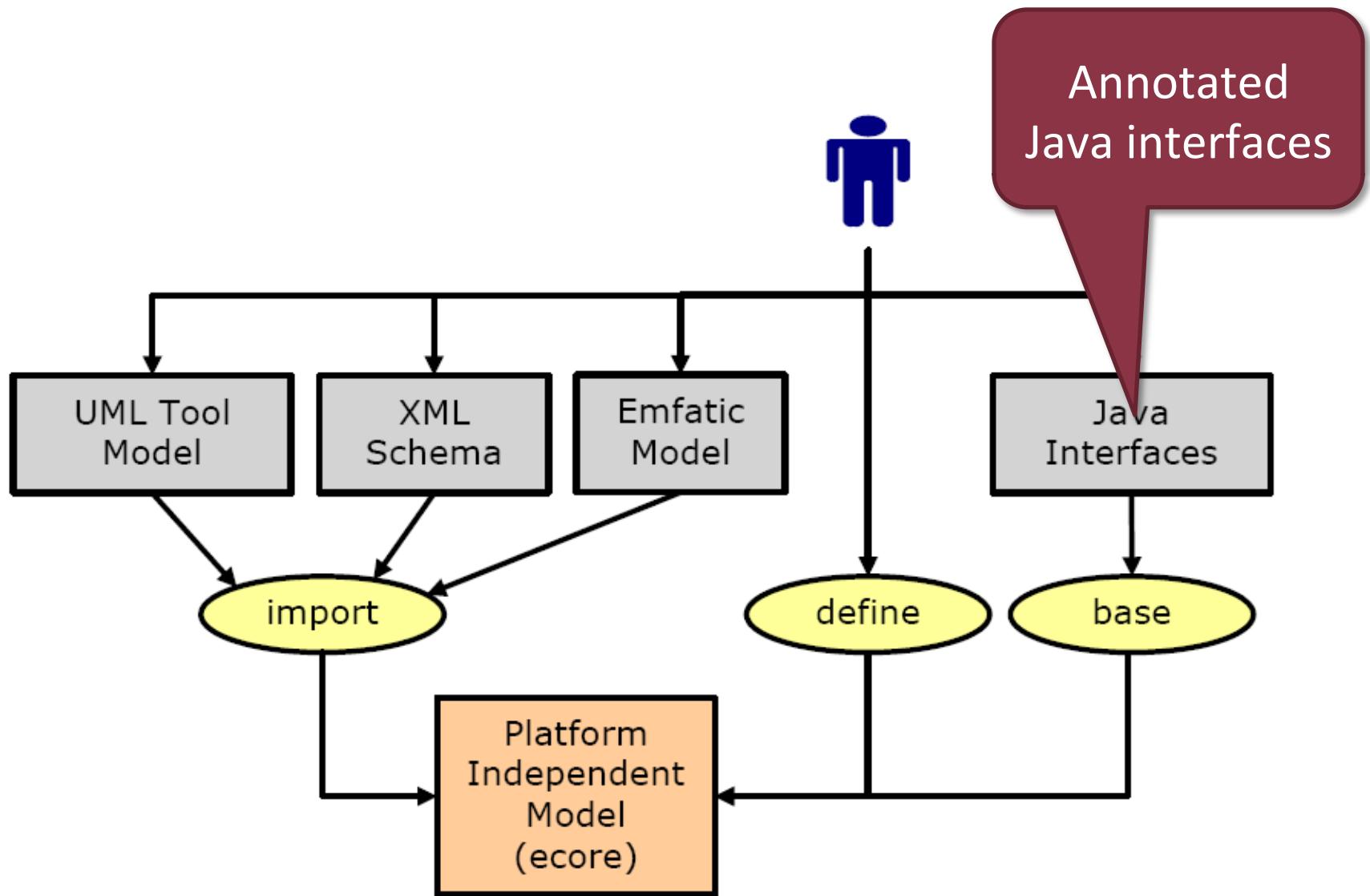
# Defining Ecore Models



# Defining Ecore Models

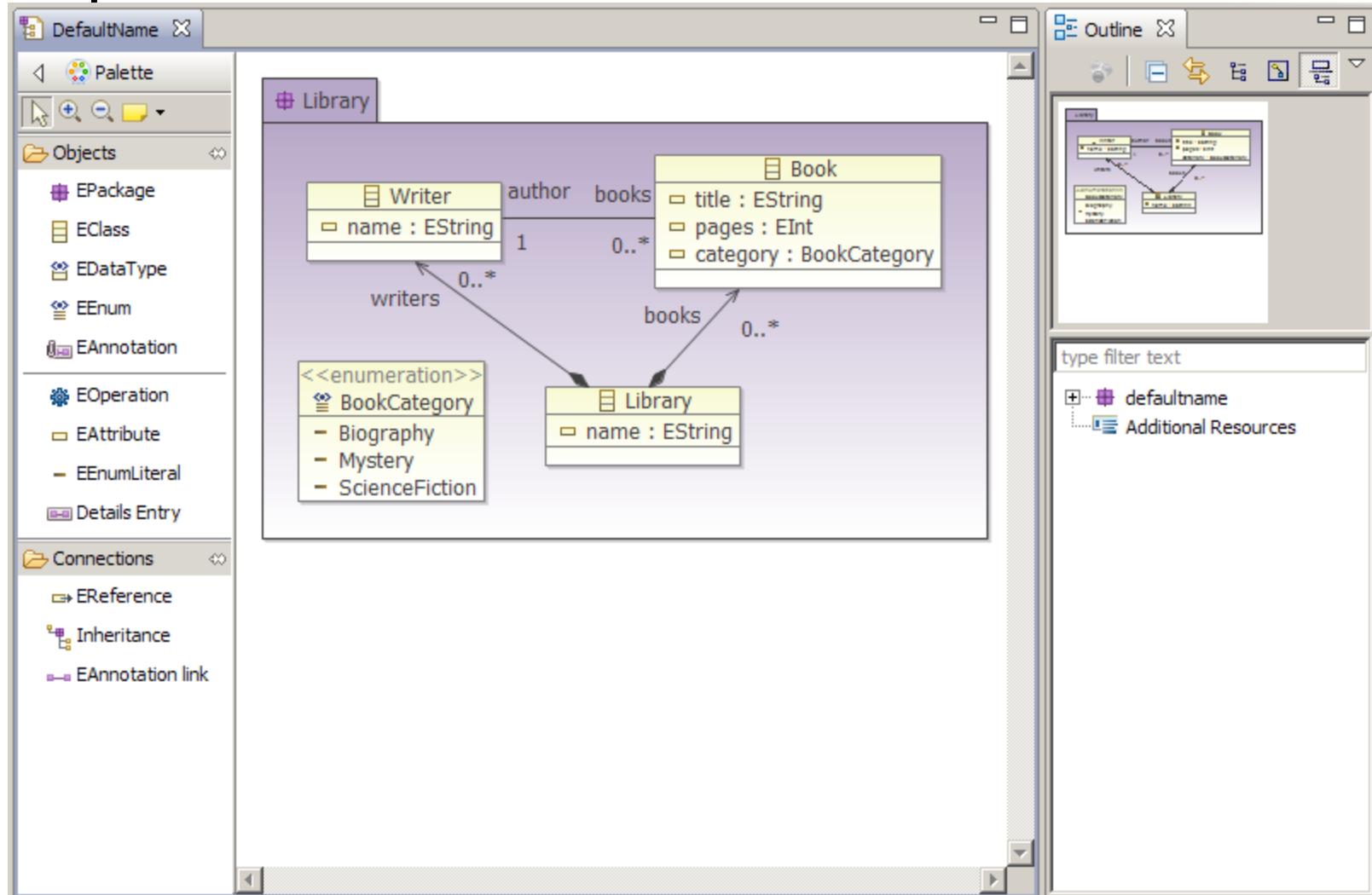


# Defining Ecore Models

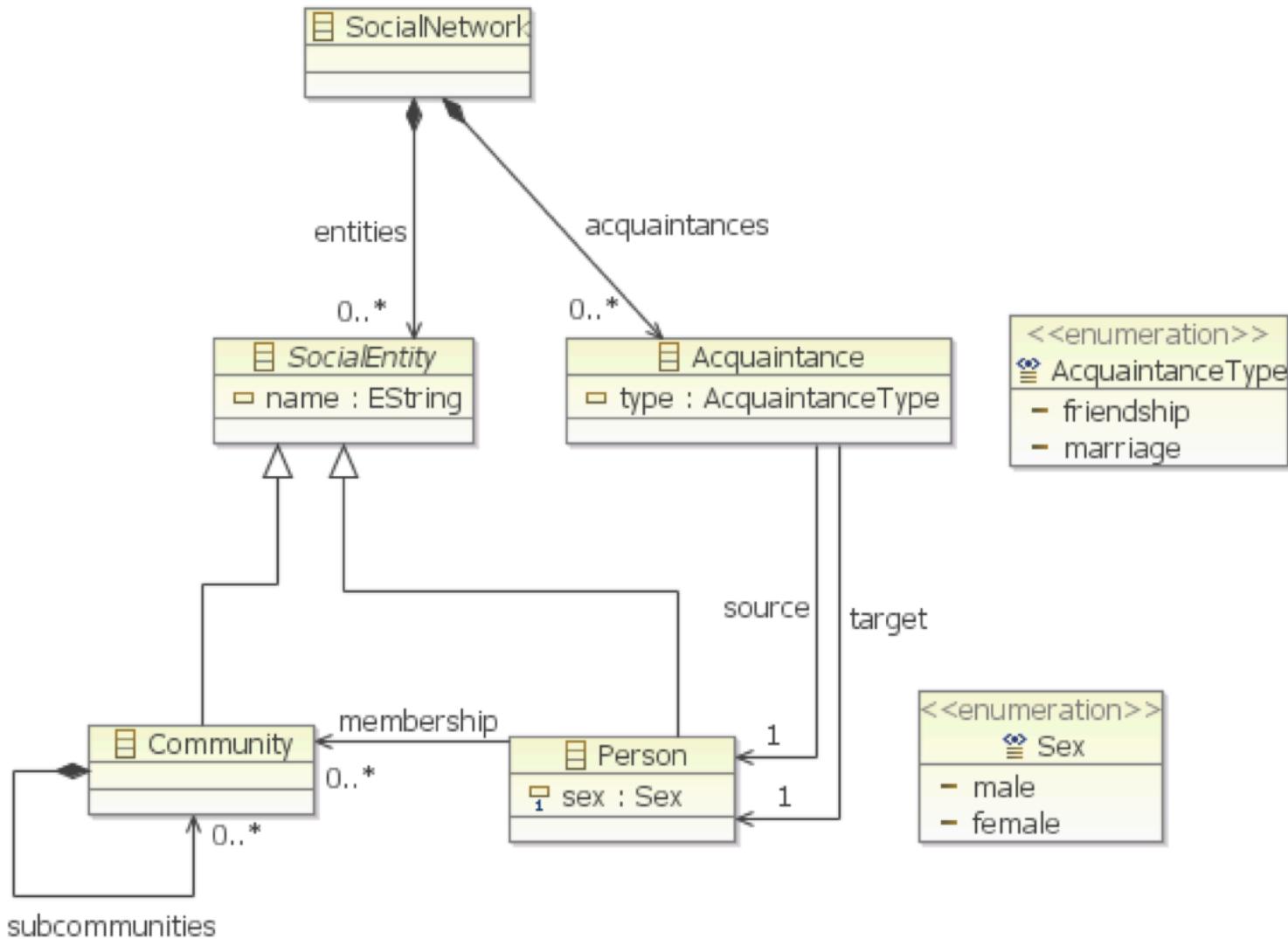


# Ecore Tools: Ecore Diagram Editor

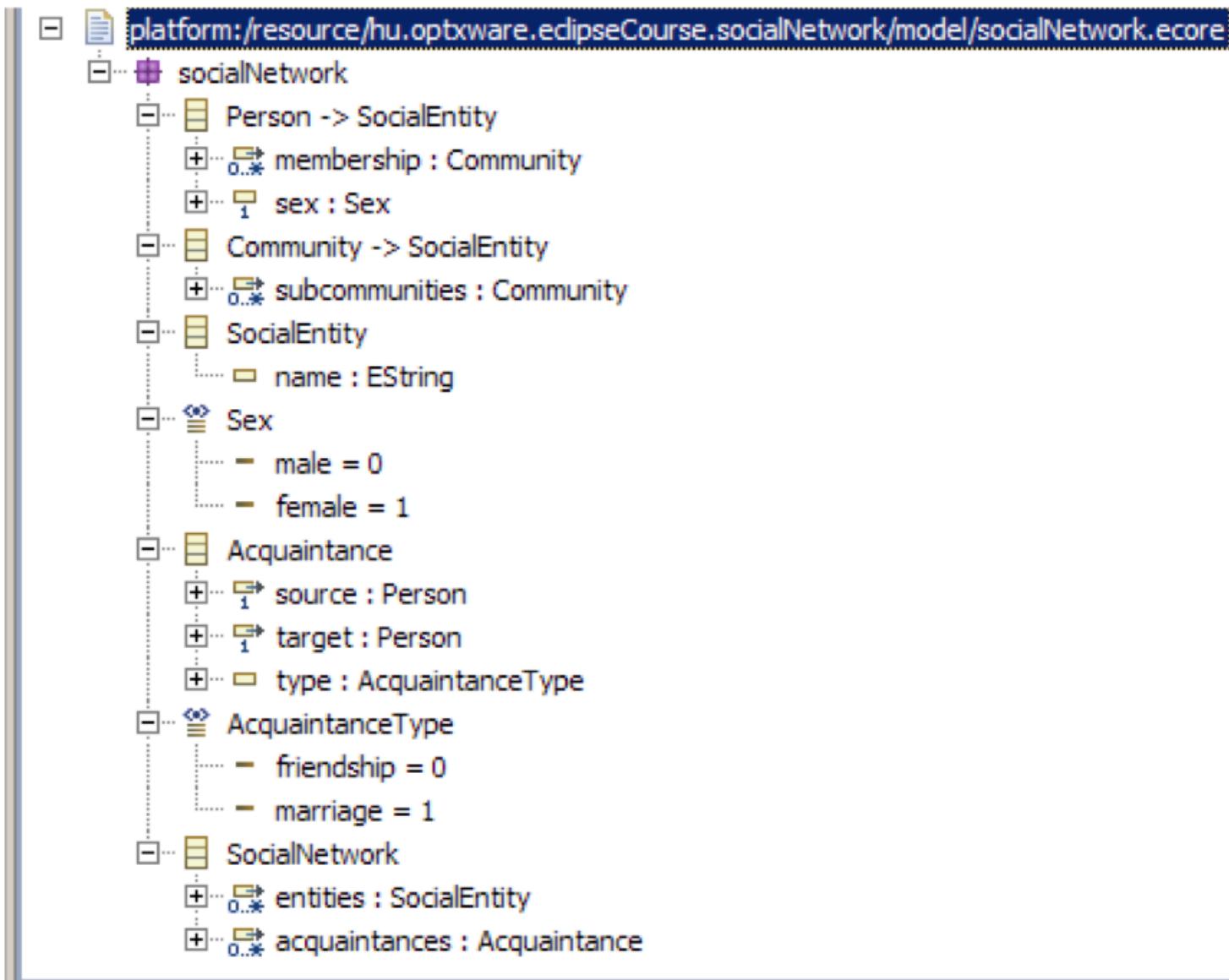
- Graphical DSL for EMF metamodels



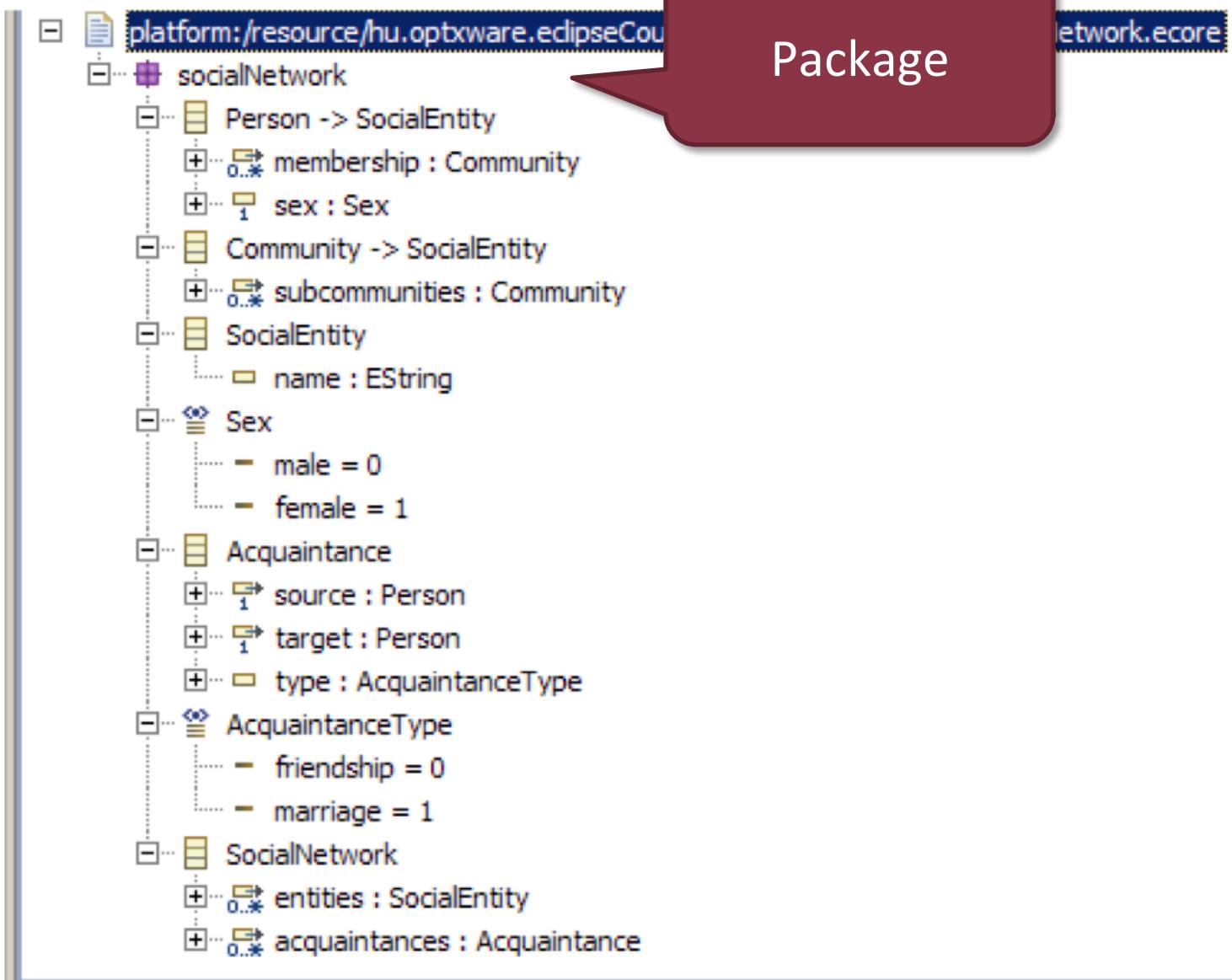
# Example: Social Network



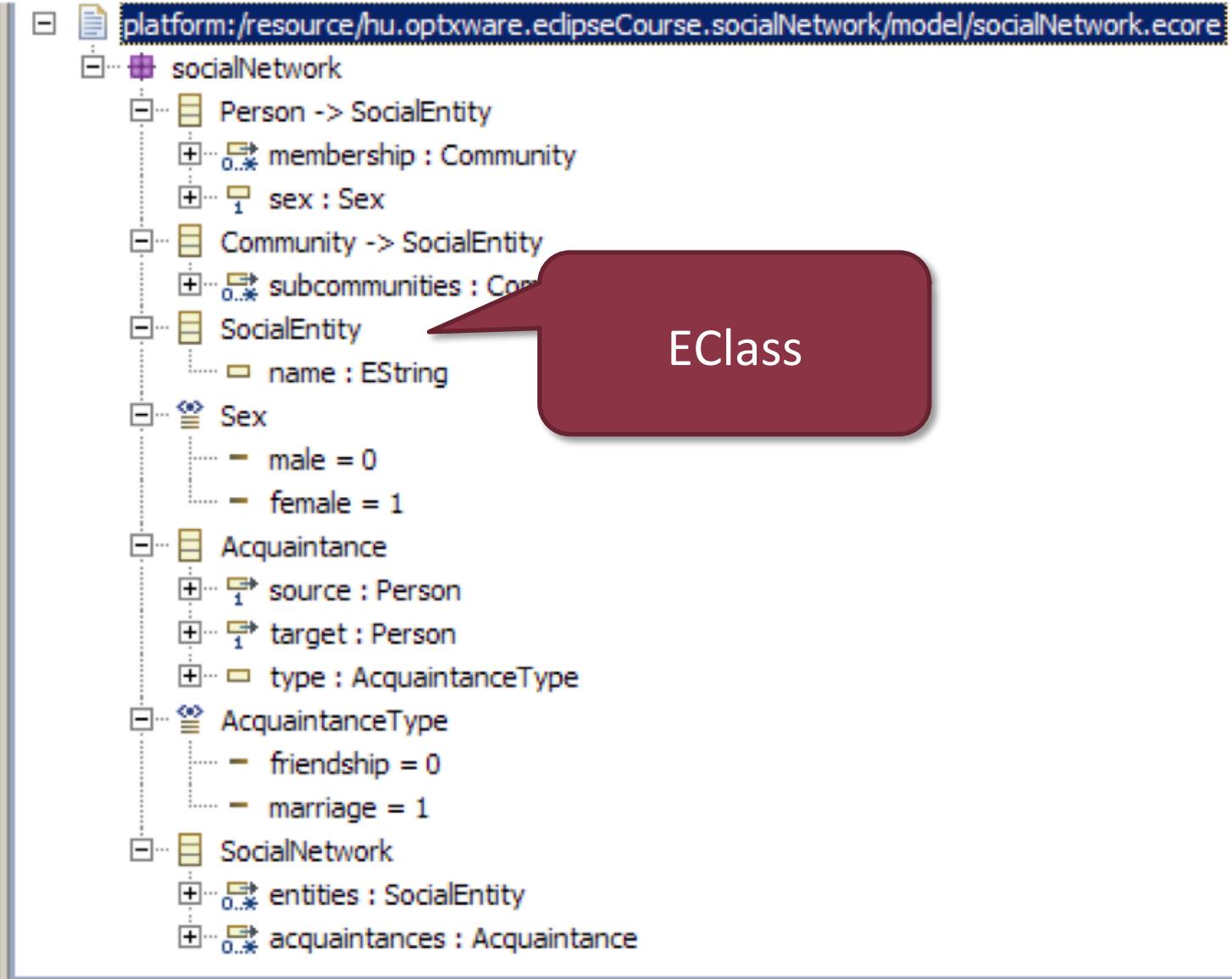
# Example: Social network



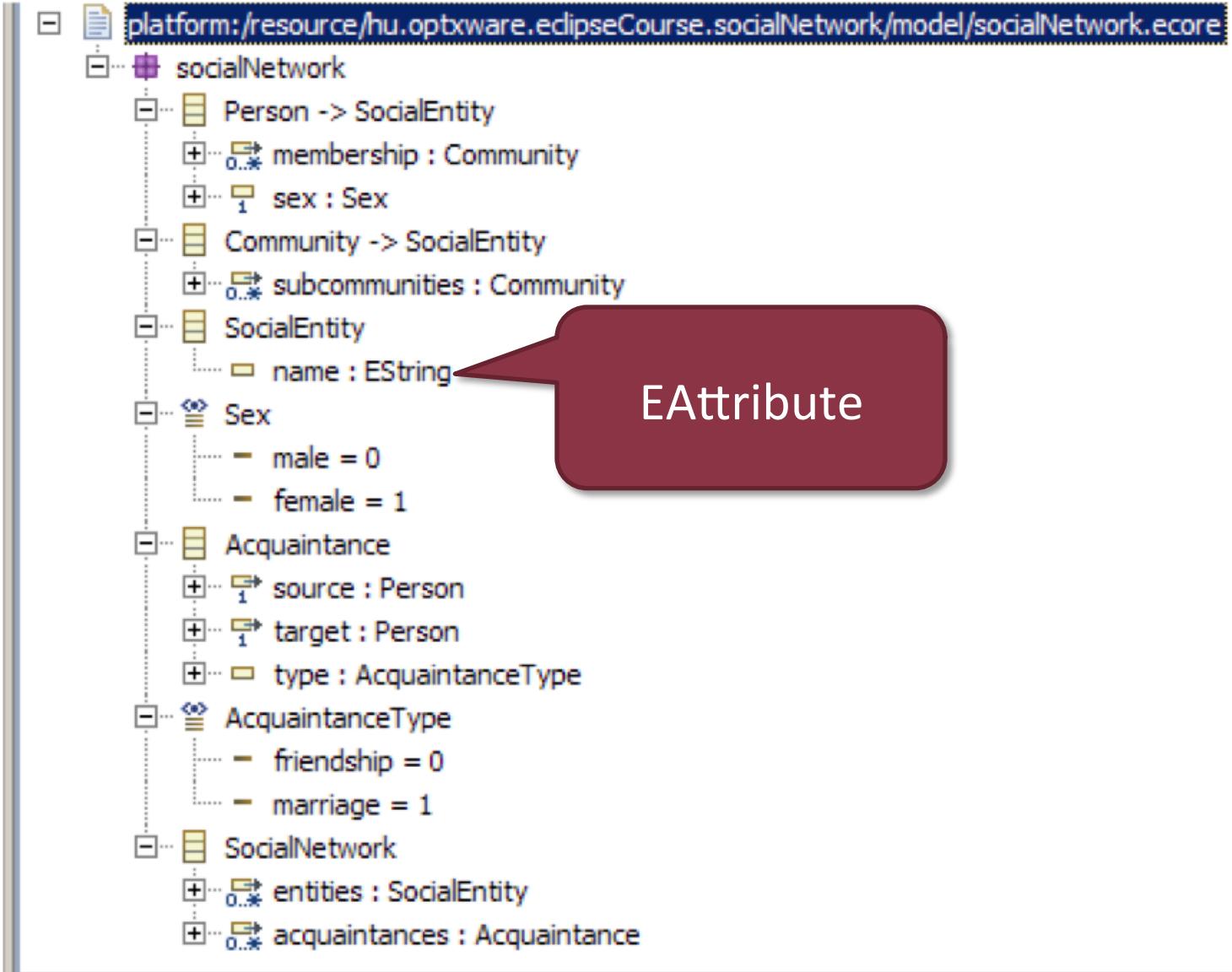
# Example: Social network



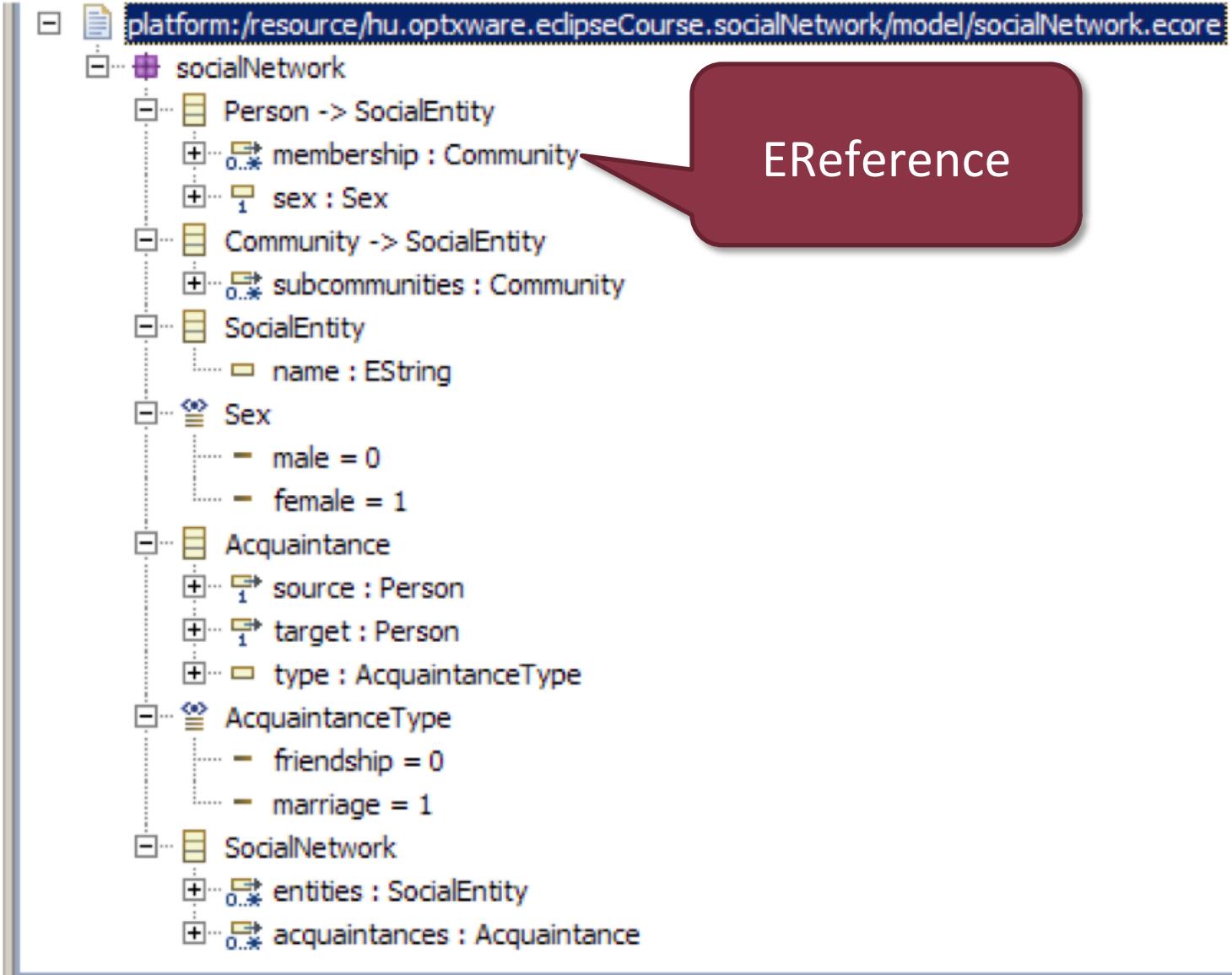
# Example: Social network



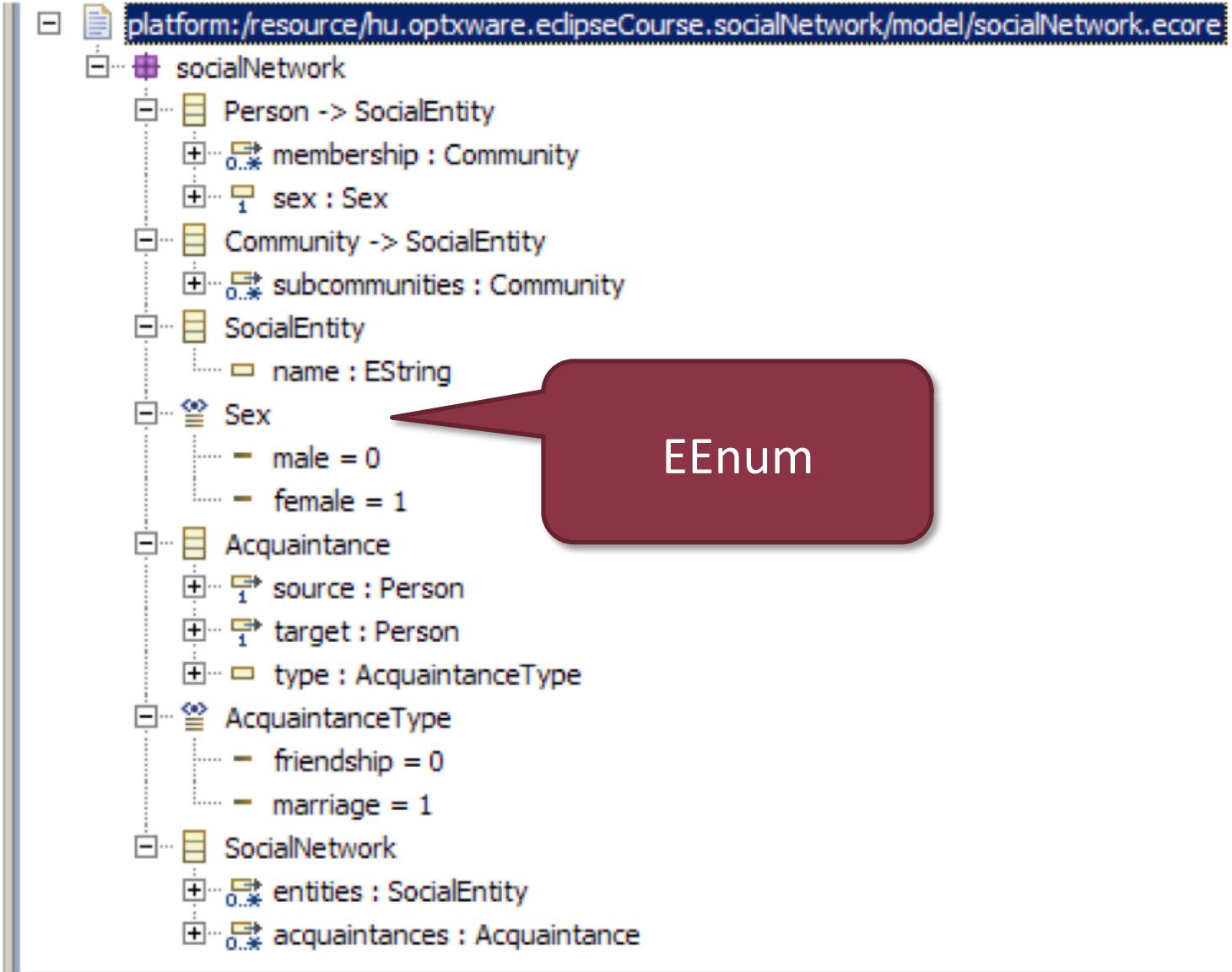
# Example: Social network



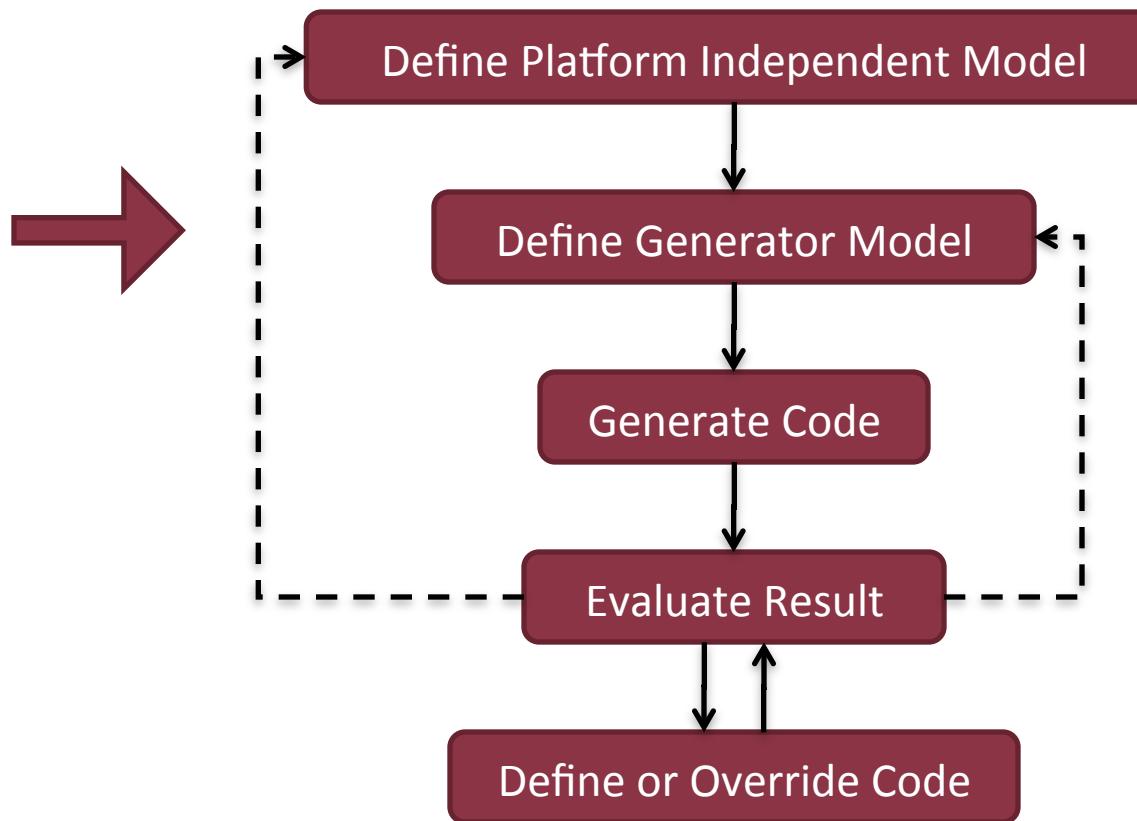
# Example: Social network



# Example: Social network



# Using EMF



# Generator model

- Providing code generation settings
- Also an EMF model
  - Refers to our Ecore metamodel
  - But uses a different metamodel
- Code generation settings
  - Java version (e.g. whether Java 5 enums are available)
  - Package and project names
  - ...

# Generator model

The screenshot shows the Eclipse IDE interface with the 'Properties' view open for a 'Social Network' generator model. The left pane displays a tree structure of generated classes and interfaces:

- Social Network
  - Person -> SocialEntity
  - Community -> SocialEntity
  - SocialEntity
  - Acquaintance
  - SocialNetwork
  - Sex
  - AcquaintanceType

The main area shows the properties of the 'Social Network' generator model. The properties listed are:

| Property                             | Value                                                |
|--------------------------------------|------------------------------------------------------|
| All                                  |                                                      |
| Bundle Manifest                      | true                                                 |
| Compliance Level                     | 5.0                                                  |
| Copyright Fields                     | false                                                |
| Copyright Text                       |                                                      |
| Language                             |                                                      |
| Model Name                           | Social Network                                       |
| Non-NLS Markers                      | false                                                |
| Runtime Compatibility                | false                                                |
| Runtime Jar                          | false                                                |
| Runtime Version                      | 2.5                                                  |
| Edit                                 |                                                      |
| Color Providers                      | false                                                |
| Creation Commands                    | true                                                 |
| Creation Icons                       | true                                                 |
| Edit Directory                       | /hu.optxware.eclipseCourse.socialNetwork.edit/src    |
| Edit Plug-in Class                   | socialNetwork.provider.SocialNetworkEditPlugin       |
| Edit Plug-in ID                      | hu.optxware.eclipseCourse.socialNetwork.edit         |
| Edit Plug-in Variables               |                                                      |
| Font Providers                       | false                                                |
| Optimized Has Children               | false                                                |
| Provider Root Extends Class          |                                                      |
| Table Providers                      | false                                                |
| Editor                               |                                                      |
| Creation Sub-menus                   | false                                                |
| Editor Directory                     | /hu.optxware.eclipseCourse.socialNetwork.editor/src  |
| Editor Plug-in Class                 | socialNetwork.presentation.SocialNetworkEditorPlugin |
| Editor Plug-in ID                    | hu.optxware.eclipseCourse.socialNetwork.editor       |
| Editor Plug-in Variables             |                                                      |
| Rich Client Platform                 | false                                                |
| Model                                |                                                      |
| Array Accessors                      | false                                                |
| Binary Compatible Reflective Methods | false                                                |
| Class Name Pattern                   |                                                      |

# Generator model

The screenshot shows the Eclipse IDE interface with the 'Properties' view open for a 'Social Network' generator model. The properties listed include:

| Property                             | Value                                                  |
|--------------------------------------|--------------------------------------------------------|
| All                                  |                                                        |
| Bundle Manifest                      | ✓ true                                                 |
| Compliance Level                     | ✓ 5.0                                                  |
| Copyright Fields                     | ✗ false                                                |
| Copyright Text                       | ✗                                                      |
| Language                             | ✗                                                      |
| Model Name                           | ✓ Social Network                                       |
| Non-NLS Markers                      | ✗ false                                                |
| Runtime Compatibility                | ✗ false                                                |
| Runtime Jar                          | ✗ false                                                |
| Runtime Version                      | ✗ 2.5                                                  |
| Edit                                 |                                                        |
| Color Providers                      | ✗ false                                                |
| Creation Commands                    | ✓ true                                                 |
| Creation Icons                       | ✓ true                                                 |
| Edit Directory                       | ✓ /hu.optxware.eclipseCourse.socialNetwork.edit/src    |
| Edit Plug-in Class                   | ✓ socialNetwork.provider.SocialNetworkEditPlugin       |
| Edit Plug-in ID                      | ✓ hu.optxware.eclipseCourse.socialNetwork.edit         |
| Edit Plug-in Variables               | ✗                                                      |
| Font Providers                       | ✗ false                                                |
| Optimized Has Children               | ✗ false                                                |
| Provider Root Extends Class          | ✗                                                      |
| Table Providers                      | ✗ false                                                |
| Editor                               |                                                        |
| Creation Sub-menus                   | ✗ false                                                |
| Editor Directory                     | ✓ /hu.optxware.eclipseCourse.socialNetwork.editor/src  |
| Editor Plug-in Class                 | ✓ socialNetwork.presentation.SocialNetworkEditorPlugin |
| Editor Plug-in ID                    | ✓ hu.optxware.eclipseCourse.socialNetwork.editor       |
| Editor Plug-in Variables             | ✗                                                      |
| Rich Client Platform                 | ✗ false                                                |
| Model                                |                                                        |
| Array Accessors                      | ✗ false                                                |
| Binary Compatible Reflective Methods | ✗ false                                                |
| Class Name Pattern                   | ✗                                                      |

Referenced Ecore  
model element  
(even multiple  
Ecore)

# Generator model

The screenshot shows the Eclipse IDE interface with the 'Properties' view open for a 'Social Network' generator model. The properties listed include:

| Property                             | Value                                                  |
|--------------------------------------|--------------------------------------------------------|
| All                                  |                                                        |
| Bundle Manifest                      | ✓ true                                                 |
| Compliance Level                     | ✗ 5.0                                                  |
| Copyright Fields                     | ✗ false                                                |
| Copyright Text                       | ✗                                                      |
| Language                             | ✗                                                      |
| Model Name                           | ✗ Social Network                                       |
| Non-NLS Markers                      | ✗ false                                                |
| Runtime Compatibility                | ✗ false                                                |
| Runtime Jar                          | ✗ false                                                |
| Runtime Version                      | ✗ 2.5                                                  |
| Edit                                 |                                                        |
| Color Providers                      | ✗ false                                                |
| Creation Commands                    | ✗ true                                                 |
| Creation Icons                       | ✗ true                                                 |
| Edit Directory                       | ✗ /hu.optxware.eclipseCourse.socialNetwork.edit/src    |
| Edit Plug-in Class                   | ✗ socialNetwork.provider.SocialNetworkEditPlugin       |
| Edit Plug-in ID                      | ✗ hu.optxware.eclipseCourse.socialNetwork.edit         |
| Edit Plug-in Variables               | ✗                                                      |
| Font Providers                       | ✗ false                                                |
| Optimized Has Children               | ✗ false                                                |
| Provider Root Extends Class          | ✗                                                      |
| Table Providers                      | ✗ false                                                |
| Editor                               |                                                        |
| Creation Sub-menus                   | ✗ false                                                |
| Editor Directory                     | ✗ /hu.optxware.eclipseCourse.socialNetwork.editor/src  |
| Editor Plug-in Class                 | ✗ socialNetwork.presentation.SocialNetworkEditorPlugin |
| Editor Plug-in ID                    | ✗ hu.optxware.eclipseCourse.socialNetwork.editor       |
| Editor Plug-in Variables             | ✗                                                      |
| Rich Client Platform                 | ✗ false                                                |
| Model                                |                                                        |
| Array Accessors                      | ✗ false                                                |
| Binary Compatible Reflective Methods | ✗ false                                                |
| Class Name Pattern                   | ✗                                                      |

General  
parameters

# Generator model

The screenshot shows the Eclipse IDE interface with the 'Properties' view open for a 'Social Network' generator model. The properties table lists various configuration options, many of which have their values set to 'false'. A red callout bubble points to the 'Edit' section of the properties table, containing entries like 'Edit Directory' and 'Edit Plug-in Class', with the text 'Edit parameters'.

| Property                             | Value                                                      |
|--------------------------------------|------------------------------------------------------------|
| All                                  |                                                            |
| Bundle Manifest                      | false true                                                 |
| Compliance Level                     | false 5.0                                                  |
| Copyright Fields                     | false                                                      |
| Copyright Text                       |                                                            |
| Language                             |                                                            |
| Model Name                           | false Social Network                                       |
| Non-NLS Markers                      | false                                                      |
| Runtime Compatibility                | false                                                      |
| Runtime Jar                          | false                                                      |
| Runtime Version                      | false 2.5                                                  |
| Edit                                 |                                                            |
| Color Providers                      | false                                                      |
| Creation Commands                    | false true                                                 |
| Creation Icons                       | false true                                                 |
| Edit Directory                       | false /hu.optxware.eclipseCourse.socialNetwork.edit/src    |
| Edit Plug-in Class                   | false socialNetwork.provider.SocialNetworkEditPlugin       |
| Edit Plug-in ID                      | false hu.optxware.eclipseCourse.socialNetwork.edit         |
| Edit Plug-in Variables               |                                                            |
| Font Providers                       | false                                                      |
| Optimized Has Children               | false                                                      |
| Provider Root Extends Class          |                                                            |
| Table Providers                      | false                                                      |
| Editor                               |                                                            |
| Creation Sub-menus                   | false                                                      |
| Editor Directory                     | false /hu.optxware.eclipseCourse.socialNetwork.editor/src  |
| Editor Plug-in Class                 | false socialNetwork.presentation.SocialNetworkEditorPlugin |
| Editor Plug-in ID                    | false hu.optxware.eclipseCourse.socialNetwork.editor       |
| Editor Plug-in Variables             |                                                            |
| Rich Client Platform                 | false                                                      |
| Model                                |                                                            |
| Array Accessors                      | false                                                      |
| Binary Compatible Reflective Methods | false                                                      |
| Class Name Pattern                   |                                                            |

# Generator model

| Property                             | Value                                             |
|--------------------------------------|---------------------------------------------------|
| All                                  |                                                   |
| Bundle Manifest                      | true                                              |
| Compliance Level                     | 5.0                                               |
| Copyright Fields                     | false                                             |
| Copyright Text                       |                                                   |
| Language                             |                                                   |
| Model Name                           | Social Network                                    |
| Non-NLS Markers                      | false                                             |
| Runtime Compatibility                | false                                             |
| Runtime Jar                          | false                                             |
| Runtime Version                      | 2.5                                               |
| Edit                                 |                                                   |
| Color Providers                      | false                                             |
| Creation Commands                    | true                                              |
| Creation Icons                       | true                                              |
| Edit Directory                       | /hu.optxware.eclipseCourse.socialNetwork.edit/src |
| Edit Plug-in Class                   | socialNetwork.provider.SocialNetworkEditPlugin    |
| Edit Plug-in ID                      | hu.optxware.eclipseCourse.socialNetwork.edit      |
| Edit Plug-in Variables               |                                                   |
| Font Providers                       | false                                             |
| Optimized Has Children               | false                                             |
| Provider Root Extends Class          |                                                   |
| Table Providers                      | false                                             |
| Editor                               |                                                   |
| Creation Sub-menus                   | false                                             |
| Editor Directory                     | /hu.optxware.eclipseCourse.socialNetwork.editor/  |
| Editor Plug-in Class                 | socialNetwork.presentation.SocialNetworkEditorPlu |
| Editor Plug-in ID                    | hu.optxware.eclipseCourse.socialNetwork.editor    |
| Editor Plug-in Variables             |                                                   |
| Rich Client Platform                 | false                                             |
| Model                                |                                                   |
| Array Accessors                      | false                                             |
| Binary Compatible Reflective Methods | false                                             |
| Class Name Pattern                   |                                                   |

Editor parameters

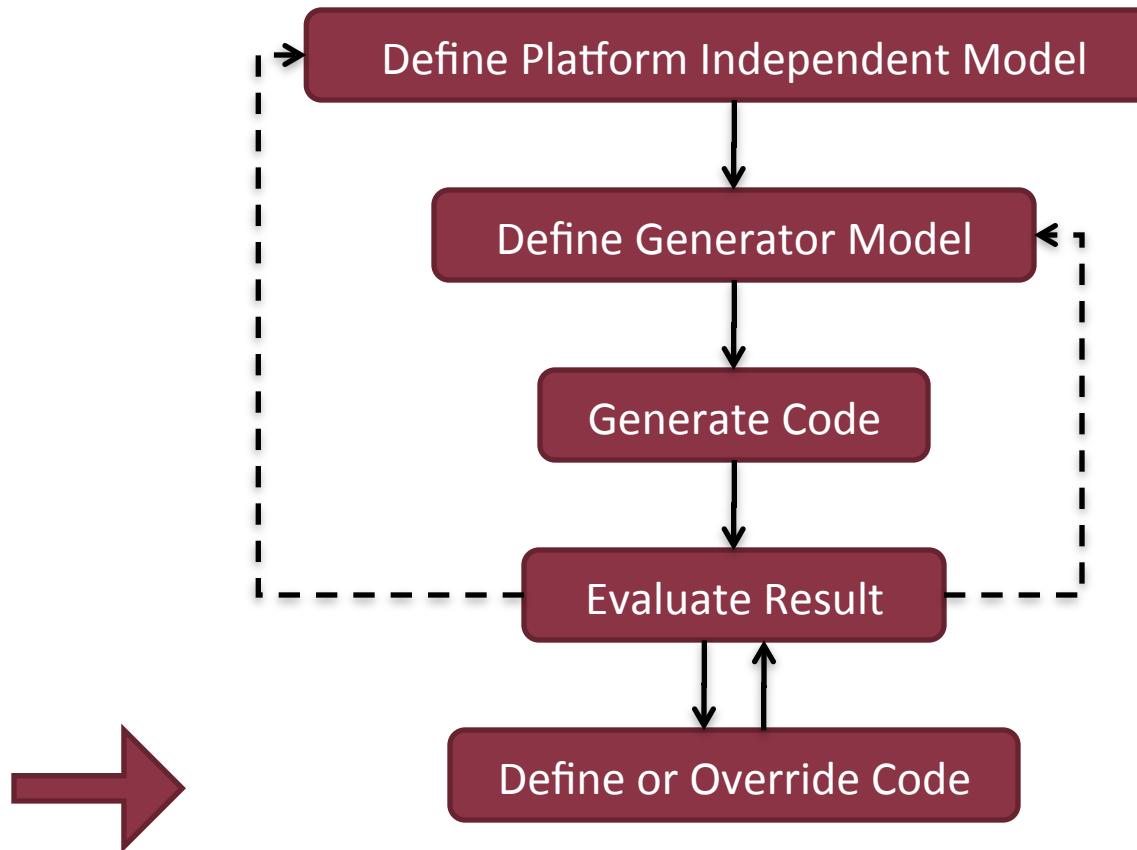
# Generator model

The screenshot shows the Eclipse IDE interface with the 'Social Network' generator model selected. The left side displays a tree view of the model's structure, including nodes for Person, Community, SocialEntity, Acquaintance, SocialNetwork, Sex, and AcquaintanceType. The right side shows the 'Properties' view for the 'Social Network' model, listing various properties and their values:

| Property                             | Value                                                |
|--------------------------------------|------------------------------------------------------|
| All                                  |                                                      |
| Bundle Manifest                      | true                                                 |
| Compliance Level                     | 5.0                                                  |
| Copyright Fields                     | false                                                |
| Copyright Text                       |                                                      |
| Language                             |                                                      |
| Model Name                           | Social Network                                       |
| Non-NLS Markers                      | false                                                |
| Runtime Compatibility                | false                                                |
| Runtime Jar                          | false                                                |
| Runtime Version                      | 2.5                                                  |
| Edit                                 |                                                      |
| Color Providers                      | false                                                |
| Creation Commands                    | true                                                 |
| Creation Icons                       | true                                                 |
| Edit Directory                       | /hu.optxware.eclipseCourse.socialNetwork.edit/src    |
| Edit Plug-in Class                   | socialNetwork.provider.SocialNetworkEditPlugin       |
| Edit Plug-in ID                      | hu.optxware.eclipseCourse.socialNetwork.edit         |
| Edit Plug-in Variables               |                                                      |
| Font Providers                       | false                                                |
| Optimized Has Children               | false                                                |
| Provider Root Extends Class          |                                                      |
| Table Providers                      | false                                                |
| Editor                               |                                                      |
| Creation Sub-menus                   | false                                                |
| Editor Directory                     | /hu.optxware.eclipseCourse.socialNetwork.editor/src  |
| Editor Plug-in Class                 | socialNetwork.presentation.SocialNetworkEditorPlugin |
| Editor Plug-in ID                    | hu.optxware.eclipseCourse.socialNetwork.editor       |
| Editor Plug-in Variables             |                                                      |
| Rich Client Platform                 | false                                                |
| Model                                |                                                      |
| Array Accessors                      | false                                                |
| Binary Compatible Reflective Methods | false                                                |
| Class Name Pattern                   |                                                      |

Model specific  
parameters

# Using EMF



# Generated EMF components

## EMF.Editor

- Simple tree based editor

## EMF.Edit

- User interface data sources
- Commands

## EMF.Model

- Model management layer
- Persistence
- Reflective API

# Generated EMF components

## EMF.Editor

- Simple tree based editor

## EMF.Edit

- User interface data sources
- Commands

## EMF.Model

- Model management layer
- Persistence
- Reflective API

# EMF.Model

- Complete implementation of the ECore metamodel
- Persistence handling
  - By default: XMI technology
  - Additionally: XSD-based XML, binary
  - Can be extended
- Model and code are similar
  - Easy to understand
  - Usually the generated code works well

# EMF.Model

- Possible extensions
  - Custom file format
    - Parser
    - See: Xtext
  - Inserting extra information into generated code
    - Avoid if possible
      - Maintainability

# Reflective and Generated API

## ■ Generated API

- Typesafe
- Simple getter/setter methods
- Preferred
  - Safe to use
  - Performs better

## ■ Reflective API

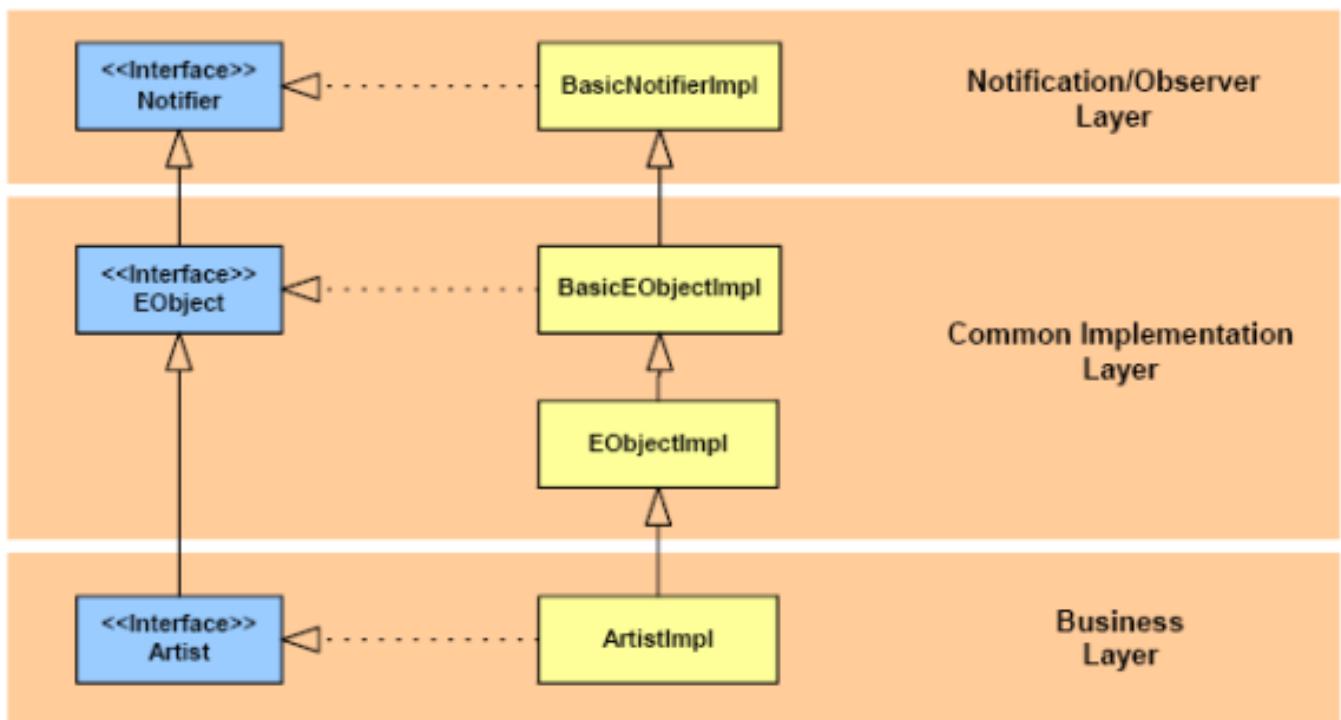
- Parameterizing with Strings or EClass instances
  - See also: dynamic languages, Java reflection
- For Generic code

# Generic or generated implementations

- Generated solution
  - Code generation based on the model
  - Unique code possible for each case
    - E.g., model-specific tree editor, GMF editor, code generator
- Generic solution
  - Same implementation used for all cases
    - E.g., serialization, Reflective Tree Editor

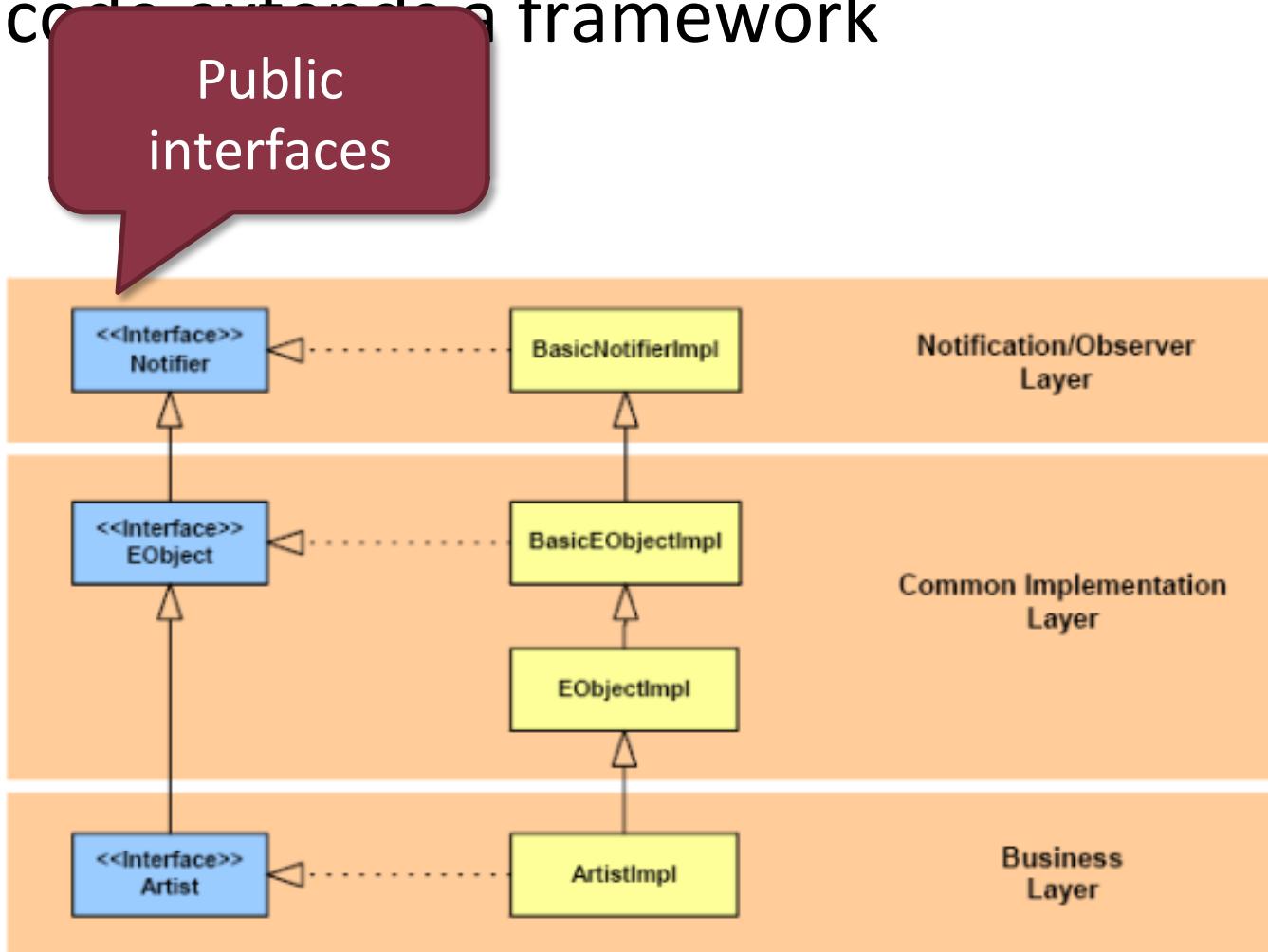
# Generated structure

- Generated code extends a framework



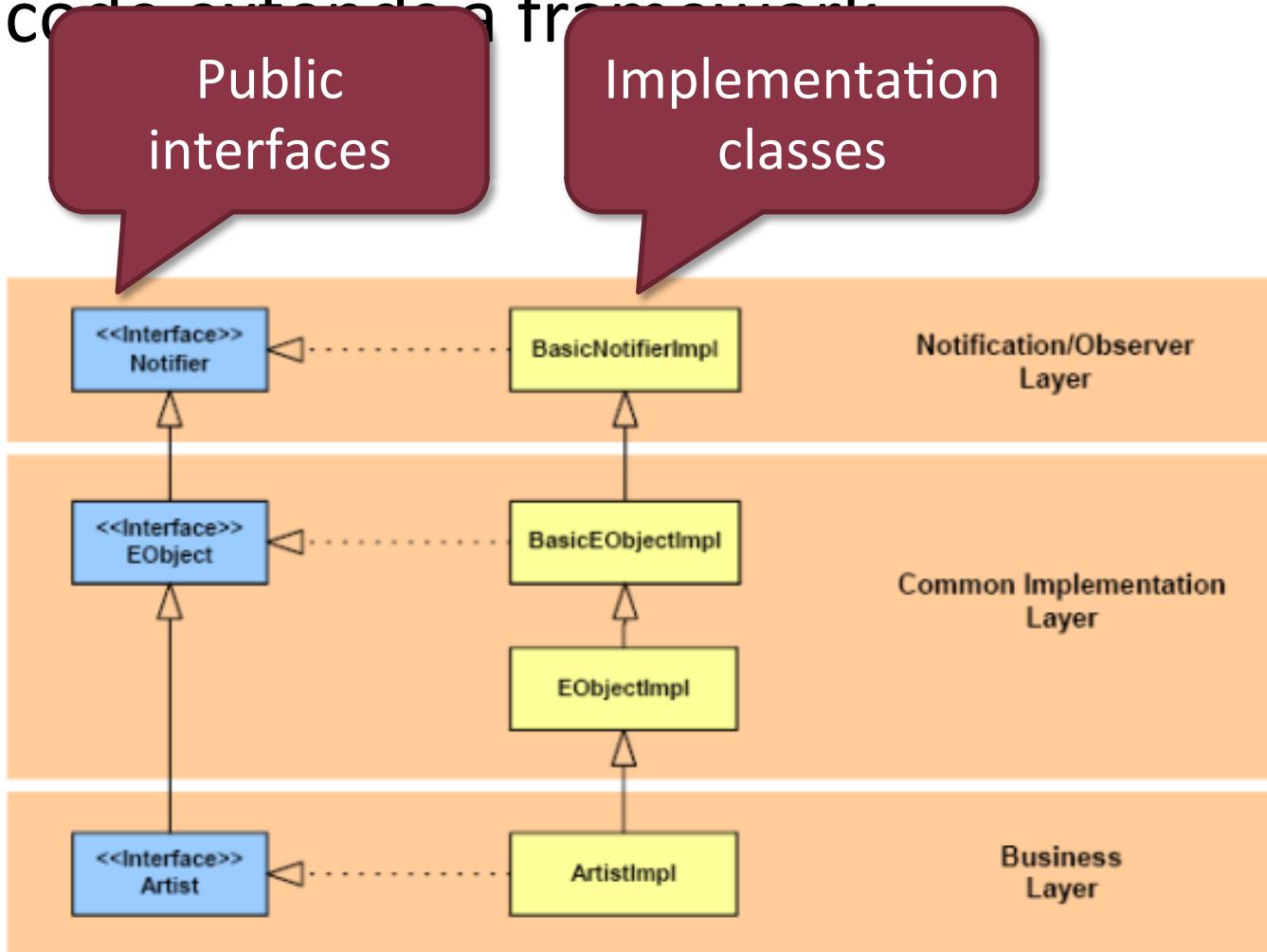
# Generated structure

- Generated code extends a framework



# Generated structure

- Generated code extends a framework



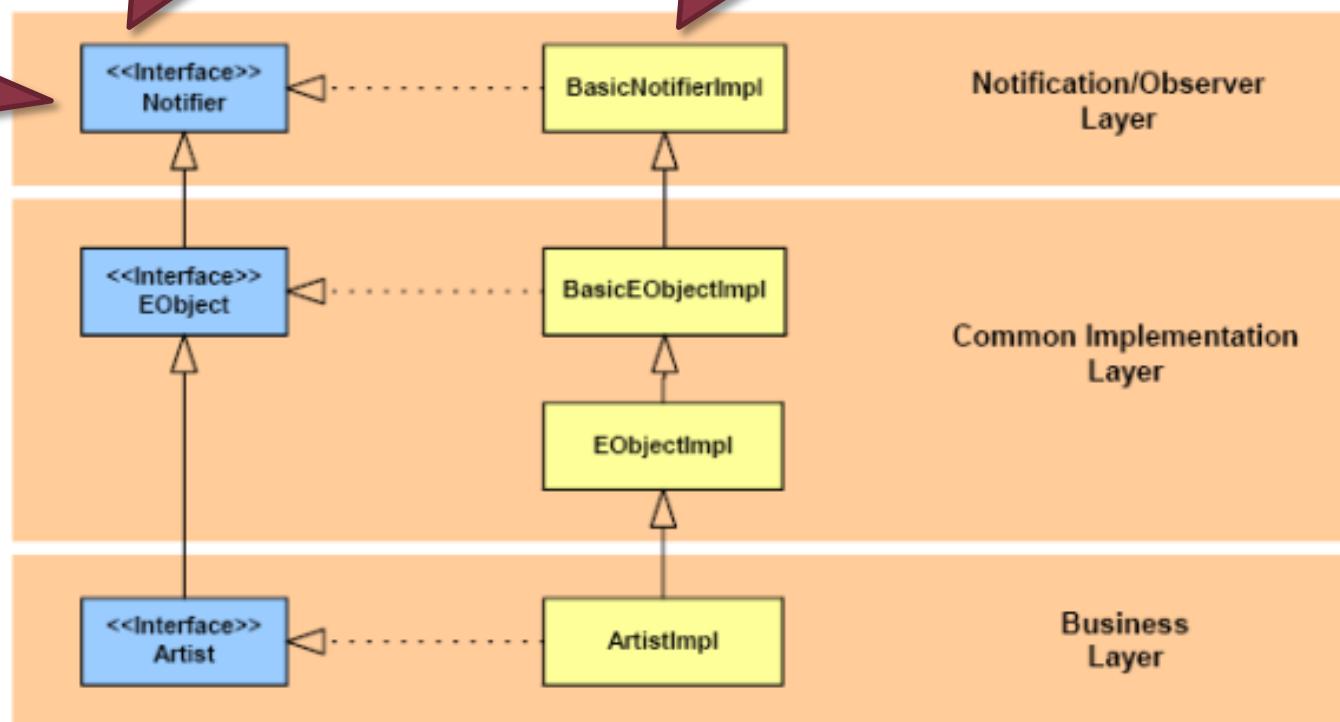
# Generated structure

- Generated code extends a framework

General notification mechanism

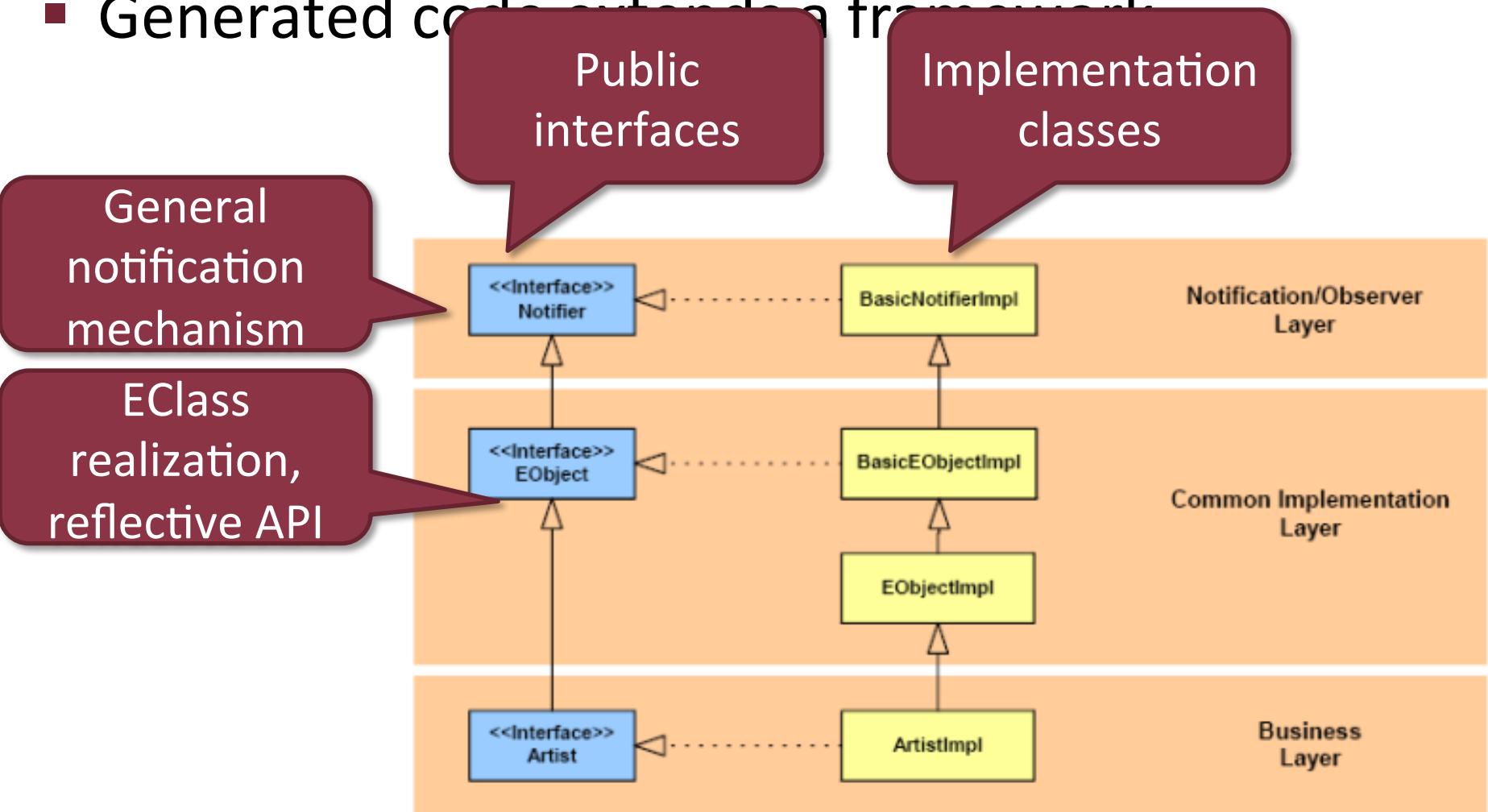
Public interfaces

Implementation classes



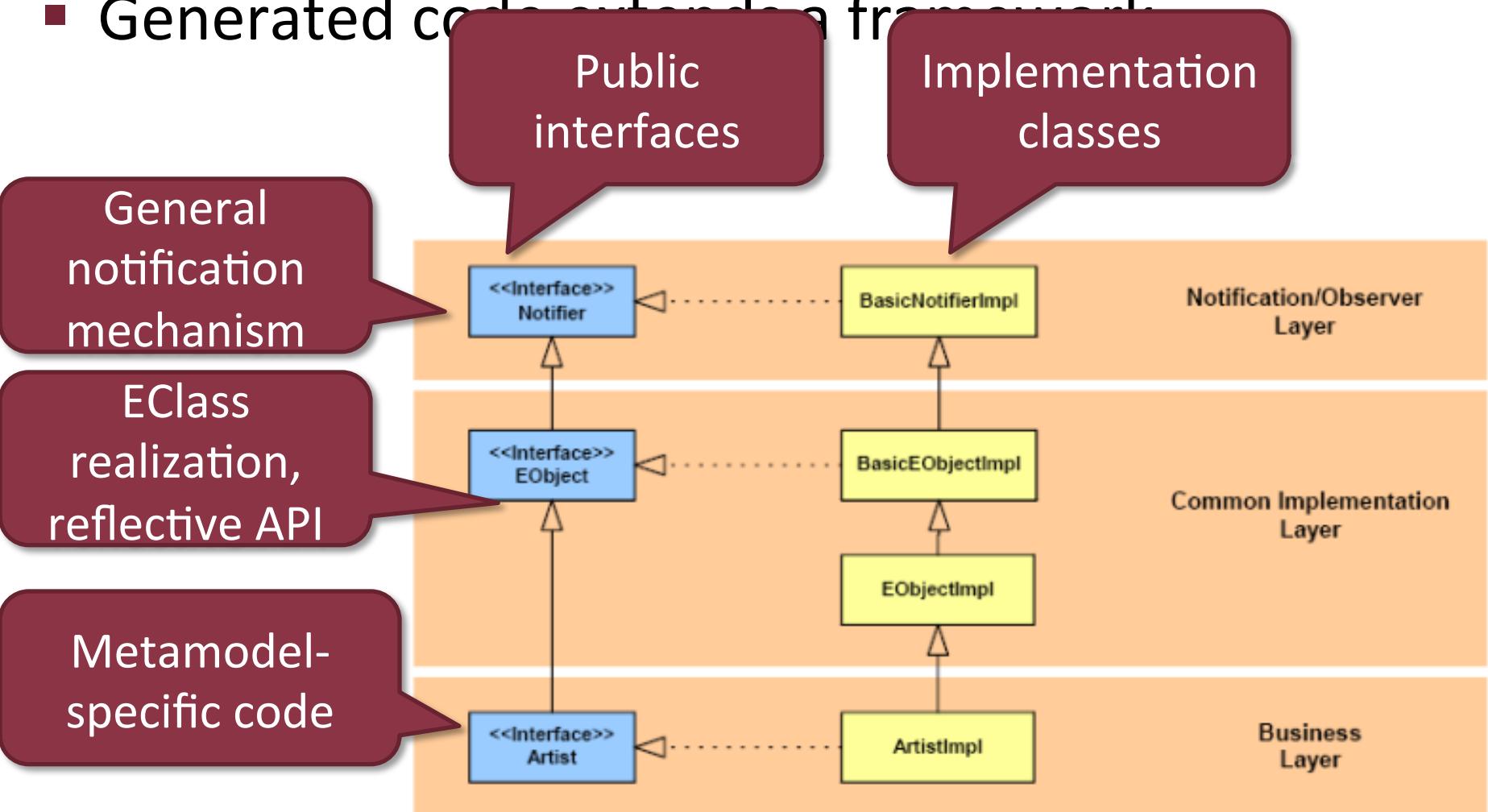
# Generated structure

- Generated code extends a framework



# Generated structure

- Generated code out of a framework



# EClass implementation

- Manages attributes and references (feature)
  - Features with “*one*” multiplicity
    - getter method
    - setter method
  - Features with “*many*” multiplicity
    - getter method returns an **editable** collection

# EReference implementation

- Type of an EReference is another EObject
- Some non-trivial cases
  - EClass can be stored in another file (Resource)
    - Reference resolution
  - Inverse references
    - Automatic synchronization

# Using EOperation

- Method declaration in EClasses
- Method body
  - ECore does not support specification
  - See Xcore project

# EFactory

- EMF object must be created with factory
- Singleton instance
  - <Package>Factory.eINSTANCE
  - <Package>Package.eINSTANCE  
    .get<csomagnév>Factory
- Concrete method for each type
  - <typename> create<typename>

# Epackage implementáció

- Singleton instance: <name>Package.eINSTANCE
- Contains EClass type literals
  - EClass get<classname>
  - EStructuralFeature  
get<classname>\_<featurename>

# Reflection

- eClass()
  - Each EObject can return its class
  - Similar to getClass() of Java
  - Generic property handling
    - eGet/eSet/eIsSet/eSet/eUnset()
- eContainer/eContents()
  - Navigating the containment hierarchy

# Reflection

- EMF Provides it
- Uses it for generic services
  - Serialization
  - Notification
  - Switch classes

# Notification

- Every model object sends change notifications
  - Observer design pattern
  - Event objects are sent
  - Notifications can be customized in generator model
- EMF provides the implementation
  - Not recommended to change it...

# Notification

- Subscription
  - `eAdapters().add(Adapter)`
  - By default it is not recursive!
- Notification sending
  - `eNotify(Notification)`
  - Seldom required manually

# Serializing EMF models

- EMF models are stored in resources
- An object is contained in a Resource instance
  - See `eResource()`
- Built-in implementation: `XMIResourceImpl`
  - Relies on XMI format

# Example: Reading and modifying a model

```
ResourceSet set = new ResourceSetImpl();
URI uri = ...;
Resource res = set.getResource(uri, true);
try {
    for (EObject root : res.getContents()) {
        //TODO Model manipulation here
    }
    res.save(new HashMap<String, String>());
} catch (IOException e) {
    // TODO Exception handling here!
    e.printStackTrace();
}
```

# Example: Reading and modifying a model

```
ResourceSet set = new ResourceSetImpl();
```

```
URI uri = ...;
```

```
Resource res = set.getResource(uri, true);
```

```
try {
```

```
    for (EObject root : res.getContents()) {  
        //TODO Model manipulation here  
    }
```

```
    res.save(new HashMap<String, String>());
```

```
} catch (IOException e) {
```

```
    // TODO Exception handling here!
```

```
    e.printStackTrace();
```

```
}
```

ResourceSet instantiation

# Example: Reading and modifying a model

```
ResourceSet set = new ResourceSetImpl();  
  
URI uri = ...;  
Resource res = set.getResource(uri);  
try {  
    for (EObject root : res.getContents()) {  
        //TODO Model manipulation here  
    }  
  
    res.save(new HashMap<String, String>());  
} catch (IOException e) {  
    // TODO Exception handling here!  
    e.printStackTrace();  
}
```

URI: model file selection

# Example: Reading and modifying a model

```
ResourceSet set = new ResourceSetImpl();
```

```
URI uri = ...;
```

```
Resource res = set.getResource(uri, true);  
try {
```

```
    for (EObject root : res.getContents())  
        //TODO Model manipulation here  
    }
```

```
    res.save(new HashMap<String, String>());  
} catch (IOException e) {  
    // TODO Exception handling here!  
    e.printStackTrace();  
}
```

getResource: on-demand loading here

# Example: Reading and modifying a model

```
ResourceSet set = new ResourceSetImpl();
```

```
URI uri = ...;
```

```
Resource res = set.getResource(uri, true);
```

```
try {
```

```
    for (EObject root : res.getContents()) {  
        //TODO Model manipulation here  
    }
```

```
    res.save(new HashMap<String, String>());  
} catch (IOException e) {  
    // TODO Exception handling here.  
    e.printStackTrace();  
}
```

Multiple model roots possible  
Type safety is not checked

# Example: Reading and modifying a model

```
ResourceSet set = new ResourceSetImpl();  
  
URI uri = ...;  
Resource res = set.getResource(uri, true);  
try {  
  
    for (EObject root : res.getContents()) {  
        //TODO Model manipulation here  
    }  
  
    res.save(new HashMap<String, String>());  
} catch (IOException e) {  
    // TODO Exception handling here  
    e.printStackTrace();  
}
```

Saving

# Example: Reading and modifying a model

```
ResourceSet set = new ResourceSetImpl();  
  
URI uri = ...;  
Resource res = set.getResource(uri, true);  
try {  
  
    for (EObject root : res.getContents()) {  
        //TODO Model manipulation here  
    }  
  
    res.save(new HashMap<String, String>());  
} catch (IOException e) {  
    // TODO Exception handling here!  
    e.printStackTrace(),  
}
```

Don't forget  
exception  
handling

# URI schema

- Resource location is described
  - By an URI
  - Creatable by the URI class
- Built-in support (extensible)
  - File (java.io.File)
    - `URI uri = URI.createFileURI("/path/to/file")`
  - Eclipse workspace file (IFile)
    - `URI.createPlatformResourceURI("projectName/foldername/filename", true)`
  - File packaged in an Eclipse plug-in
    - `URI.createPlatformPluginURI("pluginName/folderName/filename", true)`

# Serialization and the containment hierarchy

- Containment hierarchy is critical
  - Defined in the metamodel
    - Enumerating all containment reference types
- Containment hierarchy must be a forest (on the instance level)
  - Each EObject must have at most one parent
  - Via containment references all contents must be available
  - No circles allowed
  - In case of erroneous structure **serialization errors** occur
    - Referenced model element is not in the hierarchy
    - Circle in the containment hierarchy

# Important questions

1. Can a **class** be the **source** of multiple containment references?
2. Can an **object** be the **source** of multiple containment references?
3. Can a **class** be the **target** of multiple containment references?
4. Can an **object** be the **target** of multiple containment references?

# Important questions

1. Can a **class** be the **source** of multiple containment references?
2. Can an **object** be the **source** of multiple containment references?
3. Can a **class** be the **target** of multiple containment references?
4. Can an **object** be the **target** of multiple containment references?



# Important questions

1. Can a **class** be the **source** of multiple containment references? 
2. Can an **object** be the **source** of multiple containment references? 
3. Can a **class** be the **target** of multiple containment references?
4. Can an **object** be the **target** of multiple containment references?

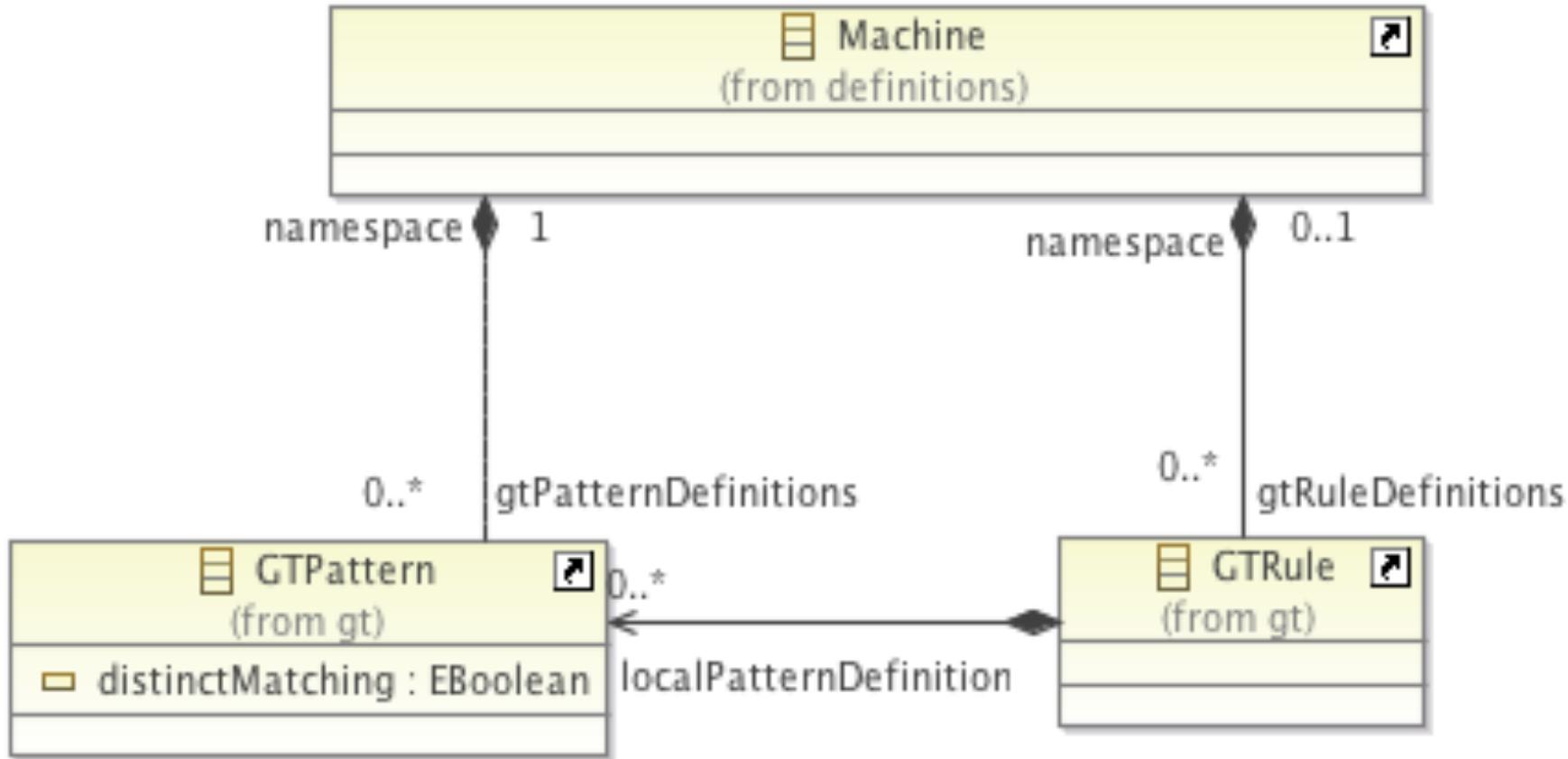
# Important questions

1. Can a **class** be the **source** of multiple containment references? 
2. Can an **object** be the **source** of multiple containment references? 
3. Can a **class** be the **target** of multiple containment references? 
4. Can an **object** be the **target** of multiple containment references?

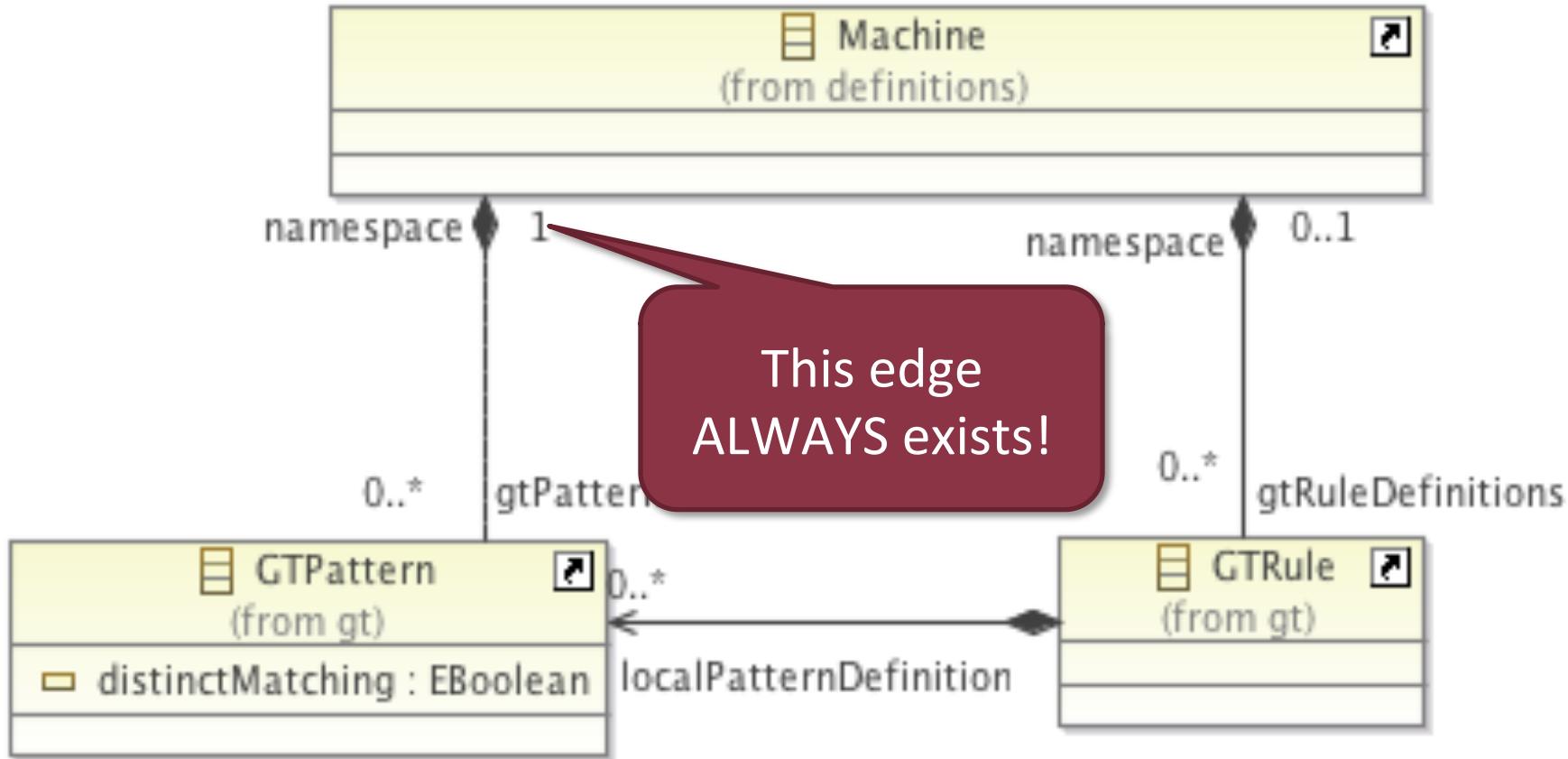
# Important questions

1. Can a **class** be the **source** of multiple containment references? 
2. Can an **object** be the **source** of multiple containment references? 
3. Can a **class** be the **target** of multiple containment references? 
4. Can an **object** be the **target** of multiple containment references? 

# What is the error in the following metamodel?



# What is the error in the following metamodel?



# A generált EMF komponensek

## EMF.Editor

- Simple tree based editor

## EMF.Edit

- User interface data sources
- Commands

## EMF.Model

- Model management layer
- Persistence
- Reflective API

# A generált EMF komponensek

## EMF.Editor

- Simple tree based editor

## EMF.Edit

- User interface data sources
- Commands

## EMF.Model

- Model management layer
- Persistence
- Reflective API

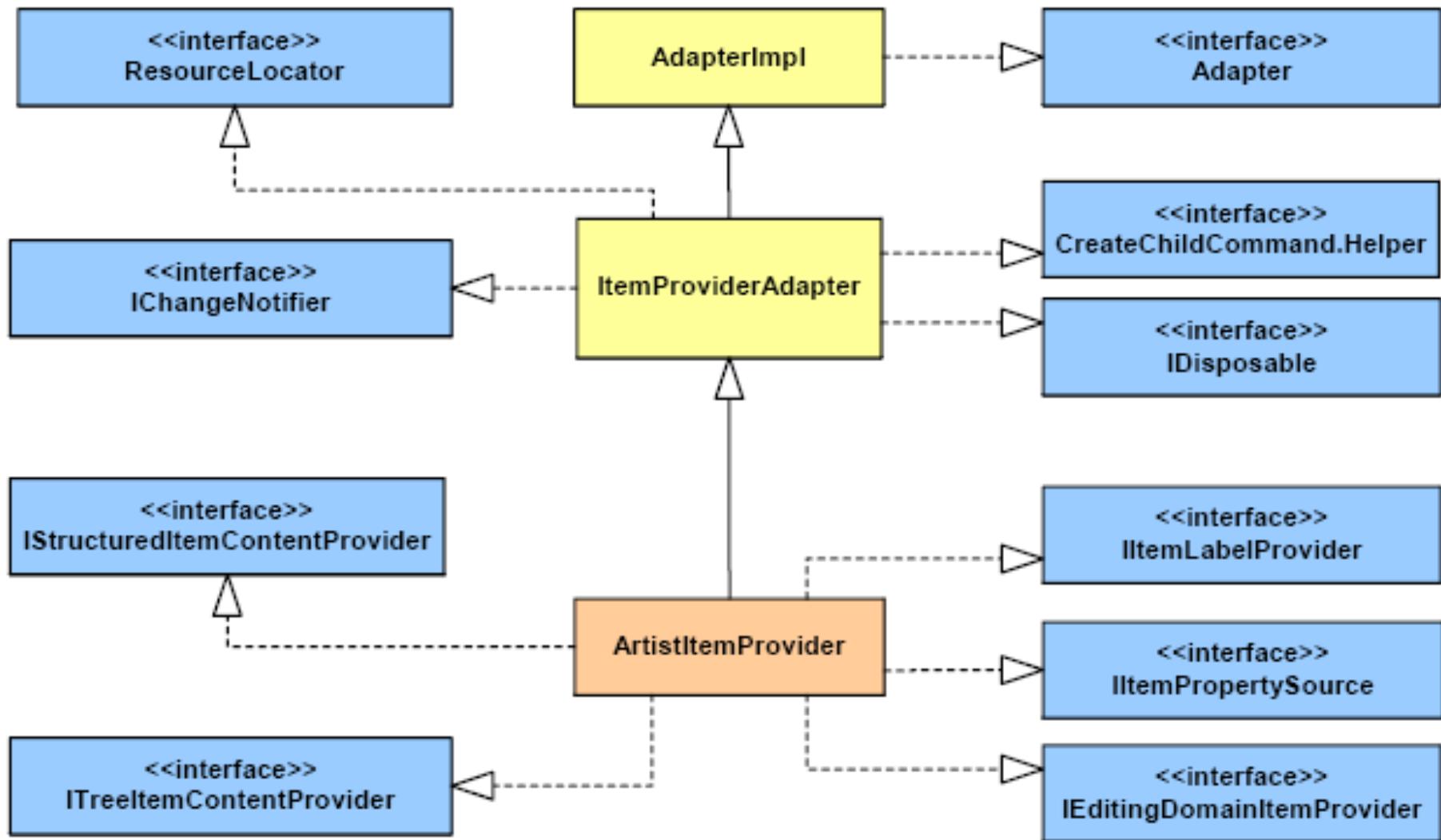
# EMF.Edit

- Goal
  - Separation of GUI and model
  - GUI-independent command implementation
- Modifications are not uncommon
  - Element provider updates (different hierarchy)
  - New command definitions

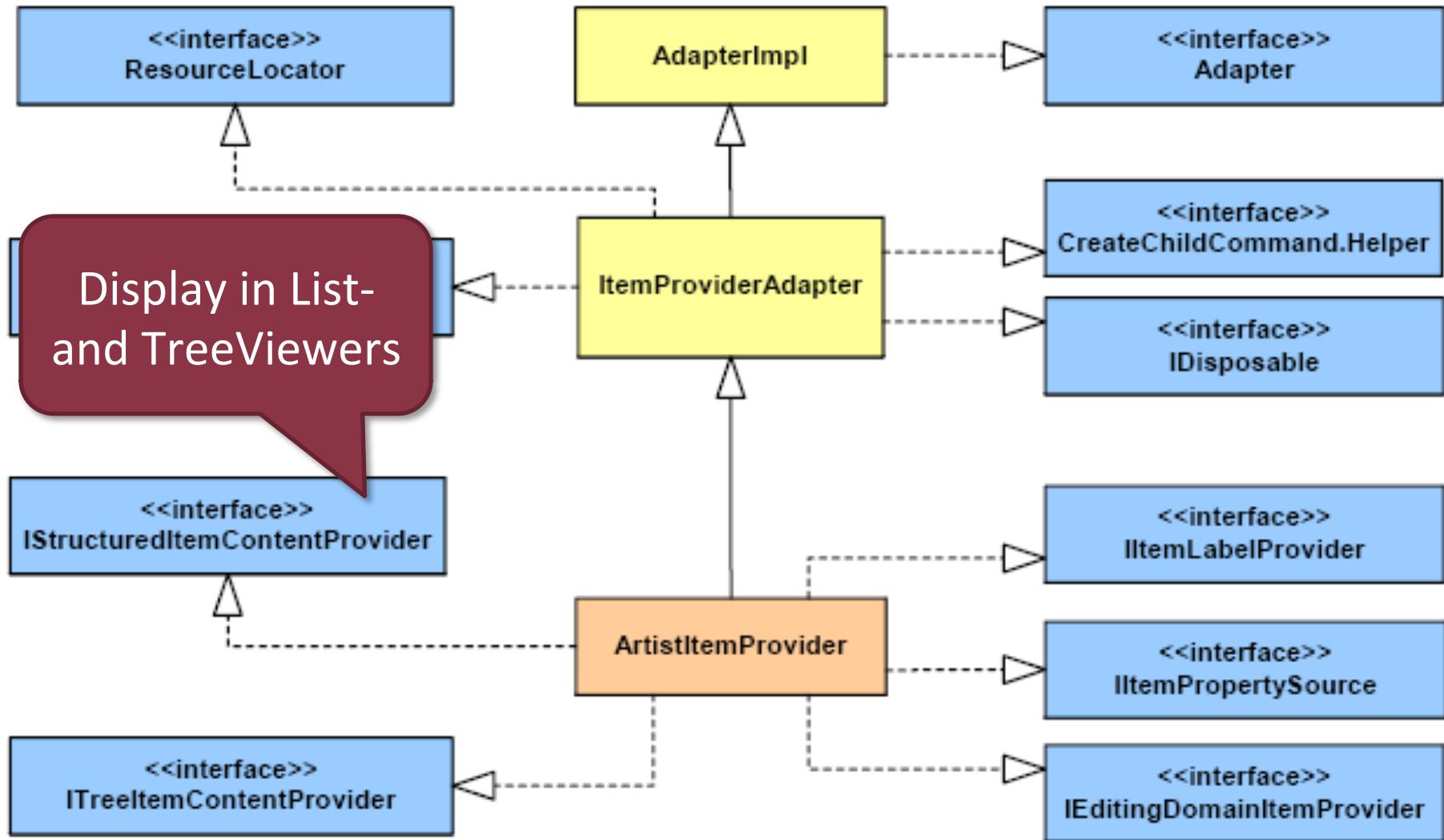
# Generator minta

- For each model object
  - An adapter is created (ItemProvider)
    - Returns child elements and possible commands
    - E.g., ArtistItemProvider
- Ancestor:
  - org.eclipse.emf.edit.provider.ItemProviderAdapter
    - Default implementation for base behaviour
    - Some parts are redefined

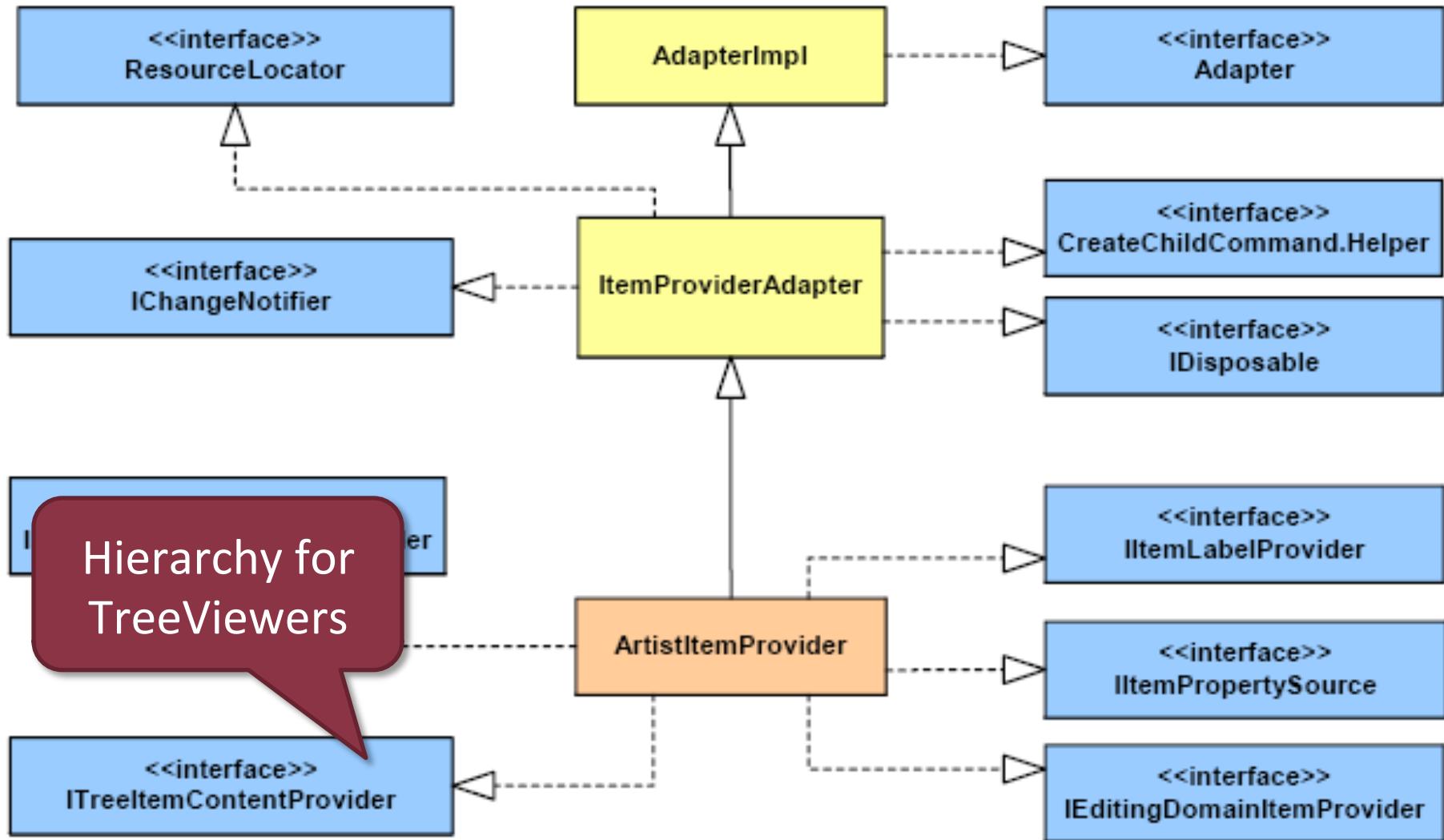
# Structure



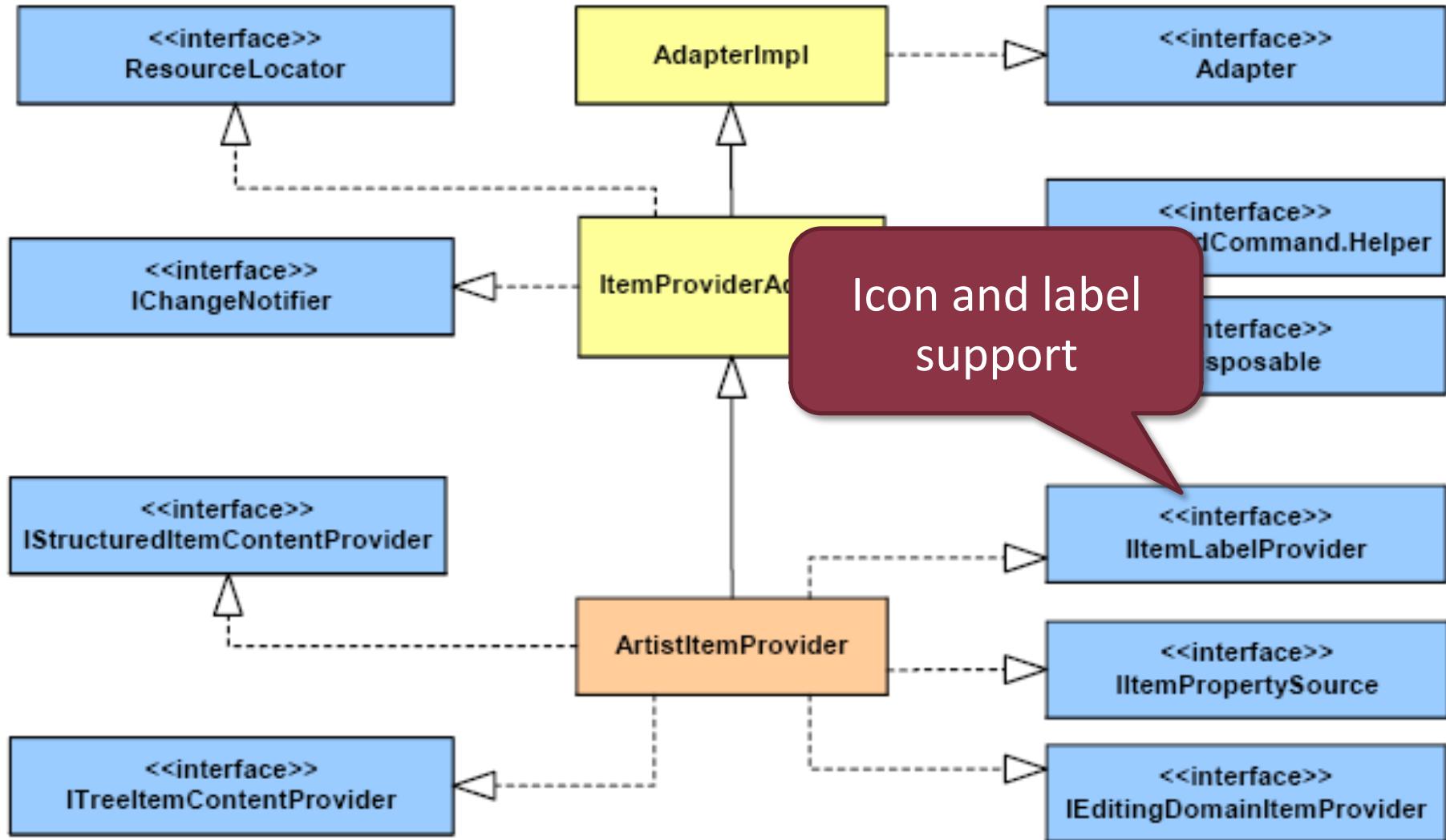
# Structure



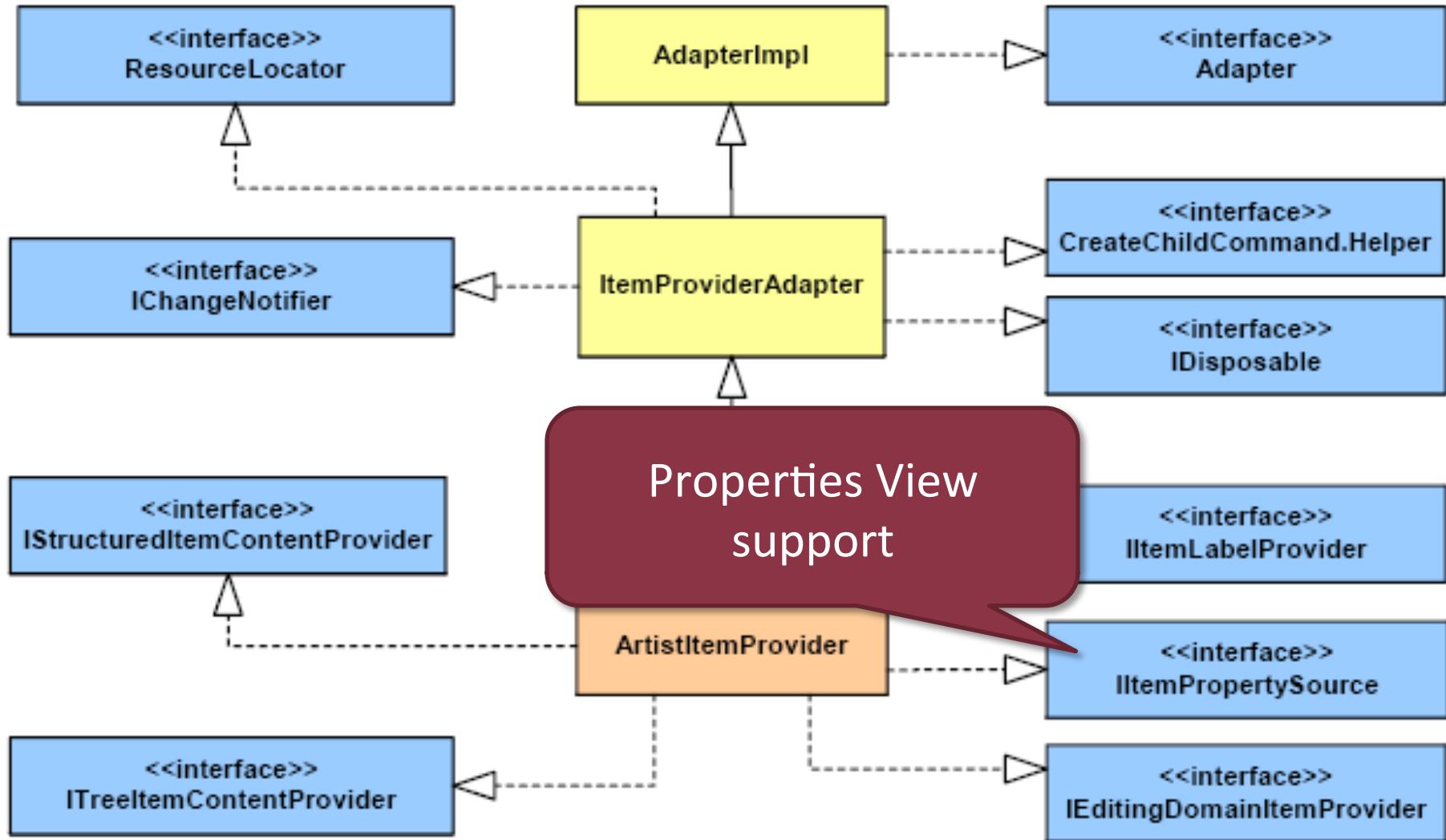
# Structure



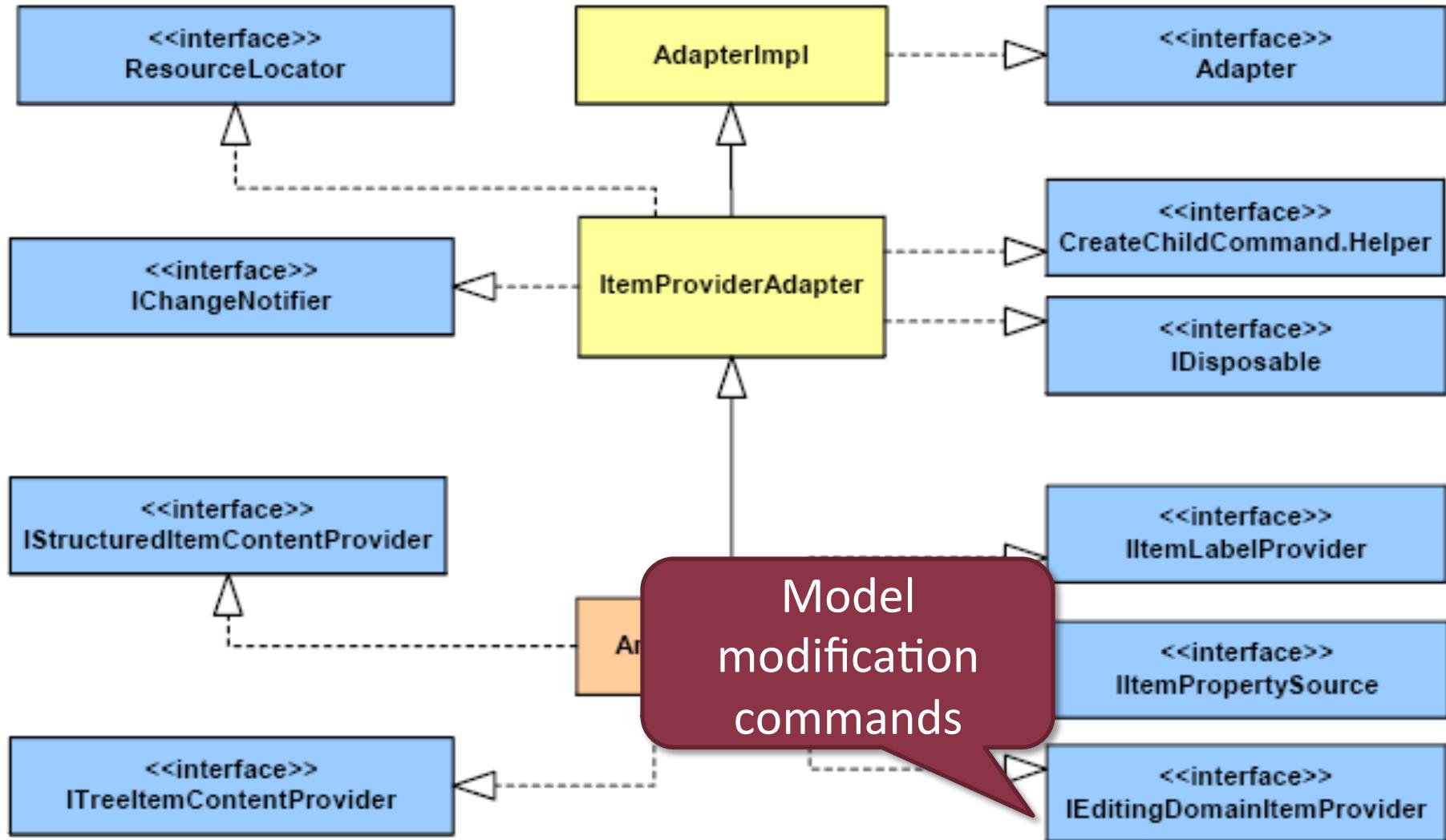
# Structure



# Structure



# Structure



# Replacing default labels

- Common customization of generated edit code
  - Simple cases can be managed in generator model
  - For complex cases update `ItemProvider.getText`
    - Updated @generated in the header to @generated NOT
    - Then provide custom implementation
      - Be mindful of notification requirements!

# Icon change

- Generator: one icon for each model element
  - Base icons generated in edit project
    - obj16: class icons
    - ctool16: child creation command
- Changes
  - Either replace the generated icon
  - Or override `ItemProvider.getImage`

# EMF.Edit commands

- Every modification happens through a command
  - Menu action
  - Attribute change
  - Drag-n-drop
  - Required for Undo/redo support
- Combination of generated and generic commands
  - EMF Common Command Framework (CCF)
  - EMF.Edit generated commands

# Commands in EMF.Edit

- ItemProvider implements `createCommand()`
- Dispatches queries to overridable methods
  - `createAddCommand()`
  - `createRemoveCommand()`
  - ...
- Changes mostly can be written in these methods

# Example EMF.Edit command: Logging added

```
public class SetArtistNameCommand extends SetCommand {  
  
    public SetArtistNameCommand(EditingDomain domain,  
EObject owner,  
EStructuralFeature feature, Object value) {  
    super(domain, owner, feature, value);  
}  
  
    public void doExecute() {  
    Artist artist = (Artist)this.owner;  
    Logger.log(MessageFormat.format(  
    "Name of artist changed from {0} to {1}",  
    artist.getName(), value.toString()));  
    super.doExecute();  
}  
  
}
```

# A generált EMF komponensek

## EMF.Editor

- Simple tree based editor

## EMF.Edit

- User interface data sources
- Commands

## EMF.Model

- Model management layer
- Persistence
- Reflective API

# A generált EMF komponensek

## EMF.Editor

- Simple tree based editor

## EMF.Edit

- User interface data sources
- Commands

## EMF.Model

- Model management layer
- Persistence
- Reflective API

# Generated editor

- Editor (based on Eclipse JFace API)
  - Tree-based
  - Provides editing commands
  - Manages workbench settings
- Menus
- Wizard (New model...)
- Plugin activator

# Generated editor

The screenshot displays a user interface for managing a social network resource set. The top-left window, titled "Resource Set", shows a tree structure with the root node "platform:/resource/Examples/default.socialnetwork". Under this root, there is a "Social Network" node, which contains four items: "Person John Doe", "Community SpongeBob Fan Club", "Person Jane Doe", and "Acquaintance friendship". The "Person John Doe" item is currently selected, highlighted with a blue border. Below the tree are five navigation tabs: Selection, Parent, List, Tree, Table, and Tree with Columns. The bottom-left window, titled "Properties", is a table showing the properties of the selected "Person John Doe" node. It has two columns: "Property" and "Value". The properties listed are Membership (Value: Community SpongeBob Fan Club), Name (Value: John Doe), and Sex (Value: male). The bottom-right window, titled "Outline", shows a hierarchical outline of the same resources, mirroring the structure seen in the Resource Set view.

| Property   | Value                        |
|------------|------------------------------|
| Membership | Community SpongeBob Fan Club |
| Name       | John Doe                     |
| Sex        | male                         |

# EMF.Editor - Problems

- Generated tree editor is only for testing
  - Tree structure is hard to understand
  - Only basic operations are supported
- Alternatives
  - GMF, Graphiti (, Sirius): graphical editors for EMF models
  - EMFText, Xtext: textual editors for EMF models

# Extension: Xcore

# Xcore: New metamodeling support

- Textual metamodel description language
  - Known shortcomings of Ecore models handled
  - Derived attributes and operations
    - Method body included
- Mostly compatible with existing tools
  - Common runtime
  - Can be generated
- Details:
  - <http://wiki.eclipse.org/Xcore>

# Xcore

The screenshot shows the Eclipse Xcore editor interface. The title bar reads "\*Library.xcore". The editor displays an Xcore metamodel for a library system:

```
package org.example.library

class Library
{
    String name
    contains Book[] books opposite library
    contains Writer[] authors opposite library
}

class Book
{
    String title
    int pages
    container Library library opposite books
    refers Writer[] authors
}

class Writer
{
    String name
    container Library library opposite authors
    refers Book[] books opposite
}
```

A tooltip window is open at the bottom right, showing "Ctrl+Space to show shortest proposals".

# Xcore

```
⊕ class Library
{
    String name
    contains Book[] books opposite library
    contains Writer[] authors opposite library
    op Book getBook(String title)
    {
        for (Book book : books)
        {
            if (title == book.title) return book
        }
        return null
    }
}

⊕ class Book..
```

# EMF - Summary

- General metamodeling framework
  - Multiple input types
  - Serialization and editing support
- Code generation
  - Customizable
  - Sane default setting
- Many users
  - Base for many Eclipse-based projects
  - Well applicable

# Related techniques

- Validation
  - Well-formedness constraint validation
- Query (not EMF-IncQuery!)
  - Query execution
- Compare
  - Structural compare (e.g. for version control)
- Teneo
  - Persists EMF models into a database
- CDO
  - Distributed, client-server EMF model handling